A Complete Normal-Form Bisimilarity for State

Dariusz Biernacki, Sergueï Lenglet, Piotr Polesiuk

A Complete Normal-Form Bisimilarity for State

Dariusz Biernacki, Sergueï Lenglet, Piotr Polesiuk

ISSN 0249-6399 ISRN INRIA/RR--9251--FR+ENG

RESEARCH REPORT
N° 9251
January 2019
Project-Team MOSEL

A Complete Normal-Form Bisimilarity for State

Dariusz Biernacki*, Sergueï Lenglet**, Piotr Polesiuk*

Project-Team MOSEL

Research Report n° 9251 — January 2019 — 32 pages

Abstract: We present a sound and complete bisimilarity for an untyped λ-calculus with higher-order local references. Our relation compares values by applying them to a fresh variable, like normal-form bisimilarity, and it uses environments to account for the evolving store. We achieve completeness by a careful treatment of evaluation contexts comprising open stuck terms. This work improves over Støvring and Lassen's incomplete environment-based normal-form bisimilarity for the $\lambda\rho$-calculus, and confirms, in relatively elementary terms, Jaber and Tabareau's result, that the state construct is discriminative enough to be characterized with a bisimilarity without any quantification over testing arguments.

Key-words: λ-calculus with state, contextual equivalence, normal-form bisimulation, completeness

* University of Wrocław, Wrocław, Poland
** Université de Lorraine, Nancy, France

RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Une bisimilarité de forme normale complète pour un calcul avec état

Résumé : Nous définissons une bisimilarité correcte et complète pour un λ-calcul non typé avec des références locales d'ordre supérieur. Notre relation compare les valeurs en leur passant comme argument une variable fraîche, comme la bisimilarité de forme normale, et utilise des environnements pour prendre en compte l'évolution de la mémoire. Nous obtenons la complétude par un traîtement méticuleux des contextes d'évaluation qui englobent les termes bloqués.

Mots-clés : λ-calcul avec état, équivalence contextuelle, bisimilarité de forme normale, complétude

1 Introduction

Two terms are contextually equivalent if replacing one by the other in a bigger program does not change the behavior of the program. The quantification over program contexts makes contextual equivalence hard to use in practice and it is therefore common to look for more effective characterizations of this relation. In a calculus with local state, such a characterization has been achieved either through *logical relations* [14,1,4], which rely on types, denotational models [9,12,5], or coinductively defined *bisimilarities* [8,17,18,16,11].

Koutavas et al. [7] argue that to be sound w.r.t. contextual equivalence, a bisimilarity for state should accumulate the tested terms in an environment to be able to try them again as the store evolves. Such *environmental bisimilarities* usually compare terms by applying them to arguments built from the environment [18,16,11], and therefore still rely on some universal quantification over testing arguments. An exception is Støvring and Lassen's bisimilarity [17], which compares terms by applying them to a fresh variable, like one would do with a *normal-form* (or *open*) bisimilarity [15,10]. Their bisimilarity characterizes contextual equivalence in a calculus with control and state, but is not *complete* in a calculus with state only: there exist equivalent terms that are not related by the bisimilarity. Jaber and Tabareau [5] go further and propose a sound and complete *Kripke Open Bisimilarity* for a calculus with local state, which also compares terms by applying them to a fresh variable, but uses notions from Kripke logical relations, namely transition systems of invariants, to reason about heaps.

In this paper, we propose a sound and complete normal-form bisimilarity for a call-by-value λ-calculus with local references which relies on environments to handle heaps. We therefore improve over Støvring and Lassen's work, since our relation is complete, by following a different, potentially simpler, path than Jaber and Tabareau, since we use environments to represent possible worlds and do not rely on any external structures such as transition systems of invariants. Moreover, we do not need types and define our relation in an untyped calculus.

We obtain completeness by treating carefully normal forms that are not values, i.e., open stuck terms of the form $E[x\ v]$. First, we distinguish in the environment the terms which should be tested multiple times from the ones that should be run only once, namely the evaluation contexts like E in the above term. The latter are kept in a separate environment that takes the form of a stack, according to the idea presented by Laird [9] and by Jagadeesan et al. [6]. Second, we relate the so-called *deferred diverging* terms [4,5], i.e., open stuck terms which hide a diverging behavior in the evaluation context E, with the regular diverging terms.

It may be worth stressing that our congruence proof is based on the machinery we have developed before [3] and is simpler than Støvring and Lassen's one, in particular in how it accounts for the extensionality of functions.

We believe that this work makes a contribution to the understanding of how one should adjust the normal-form bisimulation proof principle when the calculus under consideration becomes less discriminative, assuming that one wishes to preserve completeness of the theory. In particular, it is quite straightforward to define a complete normal-form bisimilarity for the λ-calculus with first-class continuations and global store, with no need to refer to other notions than the ones already present in the reduction semantics. Similarly, in the $\lambda\mu\rho$-calculus (continuations and local references), one only needs to introduce environments to ensure soundness of the theory, but essentially nothing more is required to obtain completeness [17]. In this article we show which new ingredients are needed when moving from these two highly expressive calculi to the corresponding, less discriminative ones—with global or local references only—that do not offer access to the current continuation.

The rest of this paper is as follows. In Section 2, we study a simple calculus with global store to see what is needed to reach completeness in that case. In particular, we show in Section 2.2 how we deal with deferred diverging terms. We remind in Section 2.3 the notion of *diacritical progress* [3] and the framework our bisimilarity and its proof of soundness are based upon. We sketch the completeness proof in Section 2.4. Section 2 paves the way for the main result of the paper, described in Section 3, where we turn to the calculus with local store. We define the bisimilarity in Section 3.2, prove its soundness and completeness in Section 3.3. We use it in Section 3.4 to prove examples taken from the literature. We conclude in Section 4, where we discuss related work and in particular compare our work to Jaber and Tabareau's.

2 Global Store

We first consider a calculus where terms share a global store and present how we deal with deferred diverging terms to get a complete bisimilarity.

2.1 Syntax, Semantics, and Contextual Equivalence

We extend the call-by-value λ-calculus with the ability to read and write a global memory. We let x, y, ... range over term variables and l range over references. A *store*, denoted by h, g, is a finite map from references to values; we write $\mathsf{dom}(h)$ for the domain of h, i.e., the set of references on which h is defined. We write $\emptyset$ for the empty store, $h \uplus g$ for the union of two stores, assuming $\mathsf{dom}(h) \cap \mathsf{dom}(g) = \emptyset$. The syntax of terms and contexts is defined as follows.

Terms:	$t, s ::= v \mid t\, t \mid l := t; t \mid !l$
Values:	$v, w ::= x \mid \lambda x.t$
Evaluation contexts:	$E, F ::= \square \mid E\ t \mid v\ E \mid l := E; t$

The term $l := t; s$ evaluates t (if possible) and stores the resulting value in l before continuing as s, while $!l$ reads the value kept in l. When writing examples and in the completeness proofs, we use natural numbers, booleans, the conditional $\mathsf{if}\ \dots\ \mathsf{then}\ \dots\ \mathsf{else}\ \dots$, local definitions $\mathsf{let} \dots \mathsf{in} \dots$, sequence ;, and unit () assuming the usual call-by-value encodings for these constructs.

A λ-abstraction $\lambda x.t$ binds x in t; we write $\mathsf{fv}(t)$ (respectively $\mathsf{fv}(E)$) for the set of free variables of t (respectively E). We identify terms up to α-conversion of their bound variables. A variable or reference is *fresh* if it does not occur in any other entities under consideration, and a store is fresh if it maps references to pairwise distinct fresh variables. A term or context is *closed* if it has no free variables. We write $\mathsf{fr}(t)$ for the set of references that occur in t.

The call-by-value semantics of the calculus is defined on *configurations* $\langle h \mid t\rangle$ such that $\mathsf{fr}(t) \subseteq \mathsf{dom}(h)$ and for all $l \in \mathsf{dom}(h)$, $\mathsf{fr}(h(l)) \subseteq \mathsf{dom}(h)$. We let c and d range over configurations. We write $t\{v/x\}$ for the usual capture-avoiding substitution of x by v in t, and we let $\int$ range over simultaneous substitutions $.\{v_1/x_1\}\dots\{v_n/x_n\}$. We write $h[l := v]$ for the operation updating the value of l to v. The reduction semantics $\rightarrow$ is defined by the following rules.

$$\langle h \mid (\lambda x.t)\ v\rangle \rightarrow \langle h \mid t\{v/x\}\rangle \qquad \langle h \mid\ !l\rangle \rightarrow \langle h \mid h(l)\rangle$$
$$\langle h \mid l := v; t\rangle \rightarrow \langle h[l := v] \mid t\rangle \qquad \langle h \mid E[t]\rangle \rightarrow \langle g \mid E[s]\rangle \text{ if } \langle h \mid t\rangle \rightarrow \langle g \mid s\rangle$$

The well-formedness condition on configurations ensures that a read operation $!l$ cannot fail. We write $\rightarrow^*$ for the reflexive and transitive closure of $\rightarrow$.

A term t of a configuration $\langle h \mid t\rangle$ which cannot reduce further is called a *normal form*. Normal forms are either values or *open-stuck terms* of the form $E[x\ v]$; closed normal forms can only be λ-abstractions. A configuration *terminates*, written $c \Downarrow$ if it reduces to a normal-form configuration; otherwise it *diverges*, written $c \Uparrow$, like configurations running $\Omega \stackrel{\mathsf{def}}{=} (\lambda x.x\ x)\ (\lambda x.x\ x)$.

Contextual equivalence equates terms behaving the same in all contexts. A substitution $\int$ closes a term t if $t\int$ is closed; it closes a configuration $\langle h \mid t\rangle$ if it closes t and the values in h.

Definition 1. *t and s are contextually equivalent, written $t \equiv s$, if for all contexts E, fresh stores h, and closing substitutions $\int$, $\langle h \mid E[t]\rangle\int \Downarrow$ iff $\langle h \mid E[s]\rangle\int \Downarrow$.*

Testing only evaluation contexts is not a restriction, as it implies the equivalence w.r.t. all contexts $\equiv_C$: one can show that $t \equiv_C s$ iff $\lambda x.t \equiv_C \lambda x.s$ iff $\lambda x.t \equiv \lambda x.s$.

2.2 Normal-Form Bisimulation

Informal presentation. Two open terms are normal-form bisimilar if their normal forms can be decomposed into bisimilar subterms. For example in the plain λ-calculus, a stuck term $E[x v]$ is bisimilar to t if t reduces to a stuck term $F[x w]$ so that respectively E, F and v, w are bisimilar when they are respectively plugged with and applied to a fresh variable.

Such a requirement is too discriminating for many languages, as it distinguishes terms that should be equivalent. For instance in plain λ-calculus, given a closed value v, $t \stackrel{\mathsf{def}}{=} x\ v$ is not normal form bisimilar to $s \stackrel{\mathsf{def}}{=} (\lambda y.x\ v)\ (x\ v)$. Indeed, $\Box$ is not bisimilar to $(\lambda y.x\ v)\ \Box$ when plugged with a fresh z: the former produces a value z while the latter reduces to a stuck term $x\,v$. However, t and s are contextually equivalent, as for all closed value w, $t\{w/x\}$ and $s\{w/x\}$

behave like $w\ v$: if $w\ v$ diverges, then they both diverges, and if $w\ v$ evaluates to some value w', then they also evaluates to w'. Similarly, $x\ v\ \Omega$ and Ω are not normal-form bisimilar (one is a stuck term while the other is diverging), but they are contextually equivalent by the same reasoning.

The terms t and s are no longer contextually equivalent in a λ-calculus with store, since a function can count how many times it is applied and change its behavior accordingly. More precisely, t and s are distinguished by the context $l := 0; (\lambda x.\square)\ \lambda z.l :=!l + 1; \mathsf{if}\ !l = 1\ \mathsf{then}\ 0\ \mathsf{else}\ \Omega$. But this counting trick is not enough to discriminate $x\ v\ \Omega$ and Ω, as they are still equivalent in a λ-calculus with store. Although $x\ v\ \Omega$ is a normal form, it is in fact always diverging when we replace x by an arbitrary closed value w, either because $w\ v$ itself diverges, or it evaluates to some w' and then $w'\ \Omega$ diverges. A stuck term which hides a diverging behavior has been called *deferred diverging* in the literature [4,5].

It turns out that being able to relate a diverging term to a deferred diverging term is all we need to change from the plain λ-calculus normal-form bisimilarity to get a complete equivalence when we add global store. We do so by distinguishing two cases in the clause for open-stuck terms: a configuration $\langle h \mid E[x\ v]\rangle$ is related to c either if c can reduce to a stuck configuration with related subterms, or if E is a diverging context, and we do not require anything of c. The resulting simulation is not symmetric as it relates a deferred diverging configuration with any configuration c (even converging one), but the corresponding notion of bisimulation equates such configuration only to either a configuration of the same kind or a diverging configuration such as $\langle h \mid \Omega\rangle$.

Progress. We define simulation using the notion of *diacritical progress* we developed in a previous work [2,3], which distinguishes between *active* and *passive* clauses. Roughly, passive clauses are between simulation states which should be considered equal, while active clauses are between states where actual progress is taking place. This distinction does not change the notions of bisimulation or bisimilarity, but it simplifies the soundness proof of the bisimilarity. It also allows for the definition of powerful *up-to techniques*, relations that are easier to use than bisimulations but still imply bisimilarity. For normal-form bisimilarity, our framework enables up-to techniques which respects η-expansion [3].

Progress is defined between objects called *candidate relations*, denoted by $\mathcal{R}$, $\mathcal{S}$, $\mathcal{T}$. A candidate relation $\mathcal{R}$ contains pairs of configurations, and a set of configurations written $\mathcal{R}{\uparrow}$, which we expect to be composed of diverging or deferred diverging configurations (for such relations we take $\mathcal{R}^{-1}{\uparrow}$ to be $\mathcal{R}{\uparrow}$). We extend $\mathcal{R}$ to stores, terms, values, and contexts with the following definitions.

$$\frac{\mathsf{dom}(h) = \mathsf{dom}(g) \qquad \forall l, h(l)\ \mathcal{R}^{\mathsf{v}}\ g(l)}{h\ \mathcal{R}^{\mathsf{h}}\ g} \qquad \frac{\langle h \mid t\rangle\ \mathcal{R}\ \langle h \mid s\rangle \qquad h \text{ fresh}}{t\ \mathcal{R}^{\mathsf{t}}\ s}$$

$$\frac{v\ x\ \mathcal{R}^{\mathsf{t}}\ w\ x \qquad x \text{ fresh}}{v\ \mathcal{R}^{\mathsf{v}}\ w} \qquad \frac{E[x]\ \mathcal{R}^{\mathsf{t}}\ F[x] \qquad x \text{ fresh}}{E\ \mathcal{R}^{\mathsf{c}}\ F} \qquad \frac{\langle h \mid E[x]\rangle \in \mathcal{R}{\uparrow} \qquad x, h \text{ fresh}}{E \in \mathcal{R}{\uparrow}^{\mathsf{c}}}$$

We use these extensions to define progress as follows.

Definition 2. *A candidate relation $\mathcal{R}$ progresses to $\mathcal{S}$, $\mathcal{T}$ written $\mathcal{R} \rightarrowtail \mathcal{S}, \mathcal{T}$, if $\mathcal{R} \subseteq \mathcal{S}$, $\mathcal{S} \subseteq \mathcal{T}$, and*

1. *$c \mathrel{\mathcal{R}} d$ implies*
   - *if $c \rightarrow c'$, then $d \rightarrow^* d'$ and $c' \mathrel{\mathcal{T}} d'$;*
   - *if $c = \langle h \mid v \rangle$, then $d \rightarrow^* \langle g \mid w \rangle$, $h \mathrel{\mathcal{S}^{\mathsf{h}}} g$, and $v \mathrel{\mathcal{S}^{\mathsf{v}}} w$;*
   - *if $c = \langle h \mid E[x\, v] \rangle$, then either*
     - *$d \rightarrow^* \langle g \mid F[x\, w] \rangle$, $h \mathrel{\mathcal{T}^{\mathsf{h}}} g$, $E \mathrel{\mathcal{T}^{\mathsf{c}}} F$, and $v \mathrel{\mathcal{T}^{\mathsf{v}}} w$, or*
     - *$E \in \mathcal{T}{\uparrow}^{\mathsf{c}}$.*
2. *$c \in \mathcal{R}{\uparrow}$ implies $c \neq \langle h \mid v \rangle$ for all h and v and*
   - *if $c \rightarrow c'$, then $c' \in \mathcal{T}{\uparrow}$;*
   - *if $c = \langle h \mid E[x\, v] \rangle$, then $E \in \mathcal{T}{\uparrow}^{\mathsf{c}}$.*

A normal-form simulation is a candidate relation $\mathcal{R}$ such that $\mathcal{R} \rightarrowtail \mathcal{R}, \mathcal{R}$, and a bisimulation is a candidate relation $\mathcal{R}$ such that $\mathcal{R}$ and $\mathcal{R}^{-1}$ are simulations. Normal-form bisimilarity $\approx$ is the union of all normal-form bisimulations.

We test values and contexts by applying or plugging them with a fresh variable x, and running them in a fresh store; with a global memory, the value represented by x may access any reference and assign it an arbitrary value, hence the need for a fresh store. The stores of two bisimilar value configurations must have the same domain, as it would be easy to distinguish them otherwise by testing the content of the references that would be in one store but not in the other.

The main novelty compared to usual definitions of normal-form bisimilarity [10,3] is the set of (deferred) diverging configurations used in the stuck terms clause. We detect that E in a configuration $\langle h \mid E[x\, v] \rangle$ is (deferred) diverging by running $\langle h' \mid E[y] \rangle$ where y and h' are fresh; this configuration may then diverge or evaluate to an other deferred diverging configuration $\langle h \mid E'[x\, v] \rangle$.

Like in the plain λ-calculus [3], $\mathcal{R}$ progresses towards $\mathcal{S}$ in the value clause and $\mathcal{T}$ in the others; the former is passive while the others are active. Our framework prevents some up-to techniques from being applied after a passive transition. In particular, we want to forbid the application of bisimulation up to context as it would be unsound: we could deduce that $v\, x$ and $w\, x$ are equivalent for all v and w just by building a candidate relation containing v and w.

Example 1. To prove that $\langle h \mid x\ v\ \Omega \rangle \approx \langle h \mid \Omega \rangle$ holds for all v and h, we prove that $\mathcal{R} \stackrel{\mathsf{def}}{=} \{(\langle h \mid x\ v\ \Omega \rangle, \langle h \mid \Omega \rangle), \{\langle g \mid y\ \Omega \rangle \mid y, g \text{ fresh}\}\}$ is a bisimulation. Indeed, $\langle h \mid x\ v\ \Omega \rangle$ is stuck with $\langle g \mid y\ \Omega \rangle \in \mathcal{R}{\uparrow}$ for fresh y and g, and we have $\langle g \mid y\ \Omega \rangle \rightarrow \langle g \mid y\ \Omega \rangle$. Conversely, the transition $\langle h \mid \Omega \rangle \rightarrow \langle h \mid \Omega \rangle$ is matched by $\langle h \mid x\ v\ \Omega \rangle \rightarrow^* \langle h \mid x\ v\ \Omega \rangle$ and the resulting terms are in $\mathcal{R}$.

2.3 Soundness

In this framework, proving that $\approx$ is sound is a consequence that a form of bisimulation up to context is valid, a result which itself may require to prove that other up-to techniques are valid. We distinguish the techniques which can be used in passive clauses (called *strong* up-to techniques), from the ones which cannot. An up-to technique (resp. strong up-to technique) is a function f such that $\mathcal{R} \rightarrowtail \mathcal{R}, f(\mathcal{R})$ (resp. $\mathcal{R} \rightarrowtail f(\mathcal{R}), f(\mathcal{R})$) implies $\mathcal{R} \subseteq \approx$. To show that a given f is an up-to technique, we rely on a notion of *respectfulness*, which is simpler to prove and gives sufficient conditions for f to be an up-to technique.

We briefly recall the notions we need from our previous work [2]. We extend $\subseteq$ and $\cup$ to functions argument-wise (e.g., $(f \cup g)(\mathcal{R}) = f(\mathcal{R}) \cup g(\mathcal{R})$), and given a set $\mathfrak{F}$ of functions, we also write $\mathfrak{F}$ for the function defined as $\bigcup_{f\in\mathfrak{F}} f$. We define f^ω as $\bigcup_{n\in\mathbb{N}} f^n$. We write id for the identity function on relations, and $\widehat{f}$ for $f \cup \mathsf{id}$. A function f is monotone if $\mathcal{R} \subseteq \mathcal{S}$ implies $f(\mathcal{R}) \subseteq f(\mathcal{S})$. We write $\mathcal{P}_{\mathit{fin}}(\mathcal{R})$ for the set of finite subsets of $\mathcal{R}$, and we say f is continuous if it can be defined by its image on these finite subsets, i.e., if $f(\mathcal{R}) \subseteq \bigcup_{\mathcal{S}\in\mathcal{P}_{\mathit{fin}}(\mathcal{R})} f(\mathcal{S})$. The up-to techniques we use are defined by inference rules with a finite number of premises, so they are trivially continuous.

Definition 3. *A function f evolves to g, h, written $f \rightsquigarrow g, h$, if for all $\mathcal{R}$ and $\mathcal{T}$, $\mathcal{R} \rightarrowtail \mathcal{R}, \mathcal{T}$ implies $f(\mathcal{R}) \rightarrowtail g(\mathcal{R}), h(\mathcal{T})$. A function f* strongly *evolves to g, h, written $f \rightsquigarrow_{\mathsf{s}} g, h$, if for all $\mathcal{R}$, $\mathcal{S}$, and $\mathcal{T}$, $\mathcal{R} \rightarrowtail \mathcal{S}, \mathcal{T}$ implies $f(\mathcal{R}) \rightarrowtail g(\mathcal{S}), h(\mathcal{T})$.*

Evolution can be seen as progress for functions on relations. Evolution is more restrictive than strong evolution, as it requires $\mathcal{R}$ such that $\mathcal{R} \rightarrowtail \mathcal{R}, \mathcal{T}$.

Definition 4. *A set $\mathfrak{F}$ of continuous functions is* respectful *if there exists $\mathfrak{S}$ such that $\mathfrak{S} \subseteq \mathfrak{F}$ and*

- *for all $f \in \mathfrak{S}$, we have $f \rightsquigarrow_{\mathsf{s}} \widehat{\mathfrak{S}}^\omega, \widehat{\mathfrak{F}}^\omega$;*
- *for all $f \in \mathfrak{F}$, we have $f \rightsquigarrow \widehat{\mathfrak{S}}^\omega \circ \widehat{\mathfrak{F}} \circ \widehat{\mathfrak{S}}^\omega, \widehat{\mathfrak{F}}^\omega$.*

In words, a function is in a respectful set $\mathfrak{F}$ if it evolves towards a combination of functions in $\mathfrak{F}$ after active clauses, and in $\mathfrak{S}$ after passive ones. When checking that f is regular (second case), we can use a regular function at most once after a passive clause. The (possibly empty) subset $\mathfrak{S}$ intuitively represents the strong up-to techniques of $\mathfrak{F}$. If $\mathfrak{S}_1$ and $\mathfrak{S}_2$ are subsets of $\mathfrak{F}$ which verify the conditions of the definition, then $\mathfrak{S}_1 \cup \mathfrak{S}_2$ also does, so there exists the largest subset of $\mathfrak{F}$ which satisfies the conditions, written $\mathsf{strong}(\mathfrak{F})$.

Lemma 1. *Let $\mathfrak{F}$ be a respectful set.*

- *If $f \in \mathfrak{F}$, then f is an up-to technique. If $f \in \mathsf{strong}(\mathfrak{F})$, then f is a strong up-to technique.*
- *For all $f \in \mathfrak{F}$, we have $f(\approx) \subseteq \approx$.*

Showing that f is in a respectful set $\mathfrak{F}$ is easier than proving it is an up-to technique. Besides, proving that a bisimulation up to context is respectful implies that $\approx$ is preserved by contexts thanks to the last property of Lemma 1.

The up-to techniques for the calculus with global store are given in Figure 1. The techniques subst and plug allow to prove that $\approx$ is preserved by substitution and by evaluation contexts. The remaining ones are auxiliary techniques which are used in the respectfulness proof: red relies on the fact that the calculus is

$$\frac{c \mathrel{\mathcal{R}} d \qquad v \mathrel{\mathcal{R}^{\mathsf{v}}} w}{c\{v/x\} \mathrel{\mathsf{subst}(\mathcal{R})} d\{w/x\}} \qquad \frac{c \in \mathcal{R}\!\uparrow}{c\{v/x\} \in \mathsf{subst}(\mathcal{R})\!\uparrow} \qquad \frac{\langle h \mid t\rangle \mathrel{\mathcal{R}} \langle g \mid s\rangle \qquad E \mathrel{\mathcal{R}^{\mathsf{c}}} F}{\langle h \mid E[t]\rangle \mathrel{\mathsf{plug}_{\mathsf{c}}(\mathcal{R})} \langle g \mid F[s]\rangle}$$

$$\frac{\langle h \mid t\rangle \in \mathcal{R}\!\uparrow}{\langle h \mid E[t]\rangle \in \mathsf{plug}_{\uparrow}(\mathcal{R})\!\uparrow} \qquad \frac{c \to^* c' \qquad d \to^* d' \qquad c' \mathrel{\mathcal{R}} d'}{c \mathrel{\mathsf{red}(\mathcal{R})} d}$$

$$\frac{c \in \mathcal{R}\!\uparrow}{c \mathrel{\mathsf{div}(\mathcal{R})} d} \qquad \frac{E \in \mathcal{R}\!\uparrow^{\mathsf{c}}}{\langle h \mid E[t]\rangle \in \mathsf{plugdiv}(\mathcal{R})\!\uparrow}$$

Fig. 1: Up-to techniques for the calculus with global store

deterministic to relate terms up to reduction steps. The technique div allows to relate a diverging configuration to any other configuration, while $\mathsf{plugdiv}$ states that if E is a diverging context, then $\langle h \mid E[t]\rangle$ is a diverging configuration for all h and t. We distinguish the technique $\mathsf{plug}_{\mathsf{c}}$ from $\mathsf{plug}_{\uparrow}$ to get a more fine-grained classification, as $\mathsf{plug}_{\mathsf{c}}$ is the only one which is not strong.

Lemma 2. *The set* $\mathfrak{F} \stackrel{\mathsf{def}}{=} \{\mathsf{subst}, \mathsf{plug}_m, \mathsf{red}, \mathsf{div}, \mathsf{plugdiv} \mid m \in \{\mathsf{c}, \uparrow\}\}$ *is respectful, with* $\mathsf{strong}(\mathfrak{F}) = \mathfrak{F} \setminus \{\mathsf{plug}_{\mathsf{c}}\}$.

We omit the proof, as it is similar but much simpler than for the calculus with local store of Section 3. We deduce that $\approx$ is sound using Lemma 1.

Theorem 1. *For all t, s, and fresh store h, if $\langle h \mid t\rangle \approx \langle h \mid s\rangle$, then $t \equiv s$.*

2.4 Completeness

We prove the reverse implication by building a bisimulation which contains $\equiv$.

Theorem 2. *For all t, s, if $t \equiv s$, then for all fresh stores h, $\langle h \mid t\rangle \approx \langle h \mid s\rangle$.*

Proof (Sketch). It suffices to show that the candidate $\mathcal{R}$ defined as

$$\begin{aligned}&\{(\langle h \mid t\rangle, \langle g \mid s\rangle) \mid \forall E, h_E,\ \text{closing } \textstyle\int, \langle h \uplus h_E \mid E[t]\rangle\textstyle\int \Downarrow\ \Rightarrow\ \langle g \uplus h_E \mid E[s]\rangle\textstyle\int \Downarrow\}\\ &\cup \{\langle h \mid t\rangle \mid \forall E, h_E,\ \text{closing } \textstyle\int, \langle h \uplus h_E \mid E[t]\rangle\textstyle\int \Uparrow\}\end{aligned}$$

is a simulation. We proceed by case analysis on the behavior of $\langle h \mid t\rangle$. The details are in Appendix B.1; we sketch the proof in the case when $\langle h \mid t\rangle \mathrel{\mathcal{R}} \langle g \mid s\rangle$, $t = E[x\,v]$, and E is not deferred diverging.

A first step is to show that $\langle g \mid s\rangle$ also evaluates to an open-stuck configuration with x in function position. To do so, we consider a fresh l and we define $\int$ such that $\int(y)$ sets l at 1 when it is first applied if $y = x$, and at 2 if $y \neq x$. Then $\langle h \uplus l := 0 \mid t\rangle\int$ sets l at 1, which should also be the case of $\langle g \uplus l := 0 \mid s\rangle\int$, and it is possible only if $\langle g \mid s\rangle \to^* \langle g' \mid F[x\,w]\rangle$ for some g', F, and w.

We then have to show that $E \mathrel{\mathcal{R}^{\mathsf{c}}} F$, $v \mathrel{\mathcal{R}^{\mathsf{v}}} w$, and $h \mathrel{\mathcal{R}^{\mathsf{h}}} g'$. We sketch the proof for the contexts, as the proofs for the values and the stores are similar.

Given h_f a fresh store, y a fresh variable, E' a context, $h_{E'}$ a store, $\int$ a closing substitution, we want $\langle h_f \uplus h_{E'} \mid E'[E[y]]\rangle\int \Downarrow$ iff $\langle h_f \uplus h_{E'} \mid E'[F[y]]\rangle\int \Downarrow$.

Let l be a fresh reference. Assuming $\mathsf{dom}(h) = \{l_1 \dots l_n\}$, given a term t, we write $\bigcup_i l_i := h; t$ for $l_1 := h(l_1); \dots l_n := h(l_n); t$. We define

$$\int_x \stackrel{\mathsf{def}}{=} \begin{cases} x \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 1; \bigcup_i l_i := h_f \uplus h_{E'}; \int(y)\ \mathsf{else}\ \int(x)\ a \\ z \mapsto \int'(z) \quad \text{if } z \neq x \end{cases}$$

The substitution $\int_x$ behaves like $\int$ except that when $\int_x(x)$ is applied for the first time, it replaces its argument by $\int(y)$ and sets the store to $h_f \uplus h_{E'}$. Therefore $\langle h \uplus l := 0 \mid E'[t]\rangle\int_x \to^* \langle h_f \uplus h_{E'} \uplus l := 1 \mid E'[E[y]]\rangle\int_x$, but this configuration then behaves like $\langle h_f \uplus h_{E'} \mid E'[E[y]]\rangle\int$. Similarly, $\langle g \uplus l := 0 \mid E'[s]\rangle\int_x$ evaluates to a configuration equivalent to $\langle h_f \uplus h_{E'} \mid E'[F[y]]\rangle\int$, and since $\langle h \uplus l := 0 \mid E'[t]\rangle\int_x \Downarrow$ implies $\langle g \uplus l := 0 \mid E'[s]\rangle\int_x \Downarrow$, we can conclude from there.

3 Local Store

We adapt the ideas of the previous section to a calculus where terms create their own local store. To be able to deal with local resources, the relation we define mixes principles from normal-form and environmental bisimilarities.

3.1 Syntax, Semantics, and Contextual Equivalence

In this section, the terms no longer share a global store, but instead must create local references before storing values. We extend the syntax of Section 2 with a construct to create a new reference.

Terms: $t, s ::= \dots \mid \mathsf{new}\ l := v\ \mathsf{in}\ t$

Reference creation $\mathsf{new}\ l := v\ \mathsf{in}\ t$ binds l in t; we identify terms up to α-conversion of their references. We write $\mathsf{fr}(t)$ and $\mathsf{fr}(E)$ for the set of free references of t or E, and a term or context is *reference-closed* if its set of free references is empty. Following [17] and in contrast with [4,5], references are not values, but we can still give access to a reference l by passing $\lambda x.!l$ and $\lambda x.l := x; \lambda y.y$.

As before, the semantics is defined on configurations $\langle h \mid t\rangle$ verifying $\mathsf{fr}(t) \subseteq \mathsf{dom}(h)$ and for all $l \in \mathsf{dom}(h)$, $\mathsf{fr}(h(l)) \subseteq \mathsf{dom}(h)$. We add to the rules of Section 2 the following one for reference creation.

$$\langle h \mid \mathsf{new}\ l := v\ \mathsf{in}\ t\rangle \to \langle h \uplus l := v \mid t\rangle$$

We remind that $\uplus$ is defined for disjoint stores only, so the above rule assumes that $l \notin \mathsf{dom}(h)$, which is always possible using α-conversion.

We define contextual equivalence on reference-closed terms as we expect programs to allocate their own store.

Definition 5. *Two reference-closed terms t and s are contextually equivalent, written $t \equiv s$, if for all reference-closed evaluation contexts E and closing substitutions $\int$, $\langle \emptyset \mid E[t]\rangle\int \Downarrow$ iff $\langle \emptyset \mid E[s]\rangle\int \Downarrow$.*

3.2 Bisimilarity

With local stores, an external observer no longer has direct access to the stored values. In presence of such information hiding, a sound bisimilarity relies on an *environment* to accumulate terms which should be tested in different stores [7].

Example 2. Let $f_1 \stackrel{\mathsf{def}}{=} \lambda x.\mathsf{if}\ !l = \mathsf{true\ then}\ l := \mathsf{false}; \mathsf{true\ else\ false}$ and $f_2 \stackrel{\mathsf{def}}{=} \lambda x.\mathsf{true}$. If we compare $\mathsf{new}\ l := \mathsf{true\ in}\ f_1$ and f_2 only once in the empty store, they would be seen as equivalent as they both return true, however f_1 modify its store, so running f_1 and f_2 a second time distinguishes them.

Environments generally contain only values [16], except in $\lambda\mu\rho$ [17], where plugged evaluation contexts are kept in the environment when comparing open-stuck configurations. In contrast with $\lambda\mu\rho$, our environment collects values, and we use a *stack* for registering contexts [9,6]. Unlike values, contexts are therefore tested only once, following a last-in first-out ordering. The next example shows that considering contexts repeatedly would lead to an overly-discriminating bisimilarity. For the stack discipline of testing contexts in action see Example 8 in Section 3.4.

Example 3. With the same f_1 and f_2 as in Example 2, the terms $t \stackrel{\mathsf{def}}{=} \mathsf{new}\ l := \mathsf{true\ in}\ f_1\ (x\ \lambda y.y)$ and $s \stackrel{\mathsf{def}}{=} f_2\ (x\ \lambda y.y)$ are contextually equivalent. Roughly, for all closing substitution $\int$, t and s either both diverge (if $\int(x)\ \lambda y.y$ diverges), or evaluate to true, since $\int(x)$ cannot modify the value in l. Testing $f_1\,\square$ and $f_2\,\square$ twice would discriminate them and wrongfully distinguish t and s.

Remark 1. The bisimilarity for $\lambda\mu\rho$ runs evaluation contexts several times and is still complete because of the μ operator, which, like $\mathsf{call/cc}$, captures evaluation contexts, and may then execute them several times.

We let $\mathcal{E}$ range over sets of pairs of values, and ϵ over sets of values. Similarly, we write Σ for a stack of pairs of evaluation contexts and σ for a stack of evaluation contexts. We write $\odot$ for the empty stack, :: for the operator putting an element on top of a stack, and $+\!\!+$ for the concatenation of two stacks. The projection operator π_1 transforms a set or stack of pairs into respectively a set or stack of single elements by taking the first element of each pair. A candidate relation $\mathcal{R}$ can be composed of:

- quadruples $(\mathcal{E}, \Sigma, c, d)$, written $\mathcal{E}, \Sigma \vdash c\ \mathcal{R}\ d$, meaning that c and d are related under $\mathcal{E}$ and Σ;
- quadruples $(\mathcal{E}, \Sigma, h, g)$, written $\mathcal{E}, \Sigma \vdash h\ \mathcal{R}\ g$, meaning that the elements of $\mathcal{E}$ and the top of Σ should be related when run with the stores h and g;
- triples (ϵ, σ, c), written $\epsilon, \sigma \vdash c \in \mathcal{R}{\uparrow}$, meaning that either c is (deferred) diverging, or σ is non-empty and contains a (deferred) diverging context;
- triples (ϵ, σ, h), written $\epsilon, \sigma \vdash h \in \mathcal{R}{\uparrow}$, meaning that σ is non-empty and contains a (deferred) diverging context.

Definition 6. *A candidate relation $\mathcal{R}$ progresses to $\mathcal{S}$, $\mathcal{T}$ written $\mathcal{R} \rightarrowtail \mathcal{S}, \mathcal{T}$, if $\mathcal{R} \subseteq \mathcal{S}$, $\mathcal{S} \subseteq \mathcal{T}$, and*

1. $\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d$ *implies*
   - *if* $c \to c'$, *then* $d \to^* d'$ *and* $\mathcal{E}, \Sigma \vdash c' \mathrel{\mathcal{T}} d'$;
   - *if* $c = \langle h \mid v \rangle$, *then either*
     - $d \to^* \langle g \mid w \rangle$, *and* $\mathcal{E} \cup \{(v, w)\}, \Sigma \vdash h \mathrel{\mathcal{S}} g$, *or*
     - $\Sigma \neq \odot$ *and* $\pi_1(\mathcal{E}) \cup \{v\}, \pi_1(\Sigma) \vdash h \in \mathcal{S}\!\uparrow$;
   - *if* $c = \langle h \mid E[x\, v] \rangle$, *then either*
     - $d \to^* \langle g \mid F[x\, w] \rangle$, *and* $\mathcal{E} \cup \{(v, w)\}, (E, F) :: \Sigma \vdash h \mathrel{\mathcal{S}} g$, *or*
     - $\pi_1(\mathcal{E}) \cup \{v\}, E :: \pi_1(\Sigma) \vdash h \in \mathcal{S}\!\uparrow$.
2. $\mathcal{E}, \Sigma \vdash h \mathrel{\mathcal{R}} g$ *implies*
   - *if* $v \mathrel{\mathcal{E}} w$, *then* $\mathcal{E}, \Sigma \vdash \langle h \mid v\, x \rangle \mathrel{\mathcal{S}} \langle g \mid w\, x \rangle$ *for a fresh* x;
   - *if* $\Sigma = (E, F) :: \Sigma'$, *then* $\mathcal{E}, \Sigma' \vdash \langle h \mid E[x] \rangle \mathrel{\mathcal{S}} \langle g \mid F[x] \rangle$ *for a fresh* x.
3. $\epsilon, \sigma \vdash c \in \mathcal{R}\!\uparrow$ *implies*
   - *if* $c \to c'$, *then* $\epsilon, \sigma \vdash c' \in \mathcal{T}\!\uparrow$;
   - *if* $c = \langle h \mid v \rangle$, *then* $\sigma \neq \odot$ *and* $\epsilon \cup \{v\}, \sigma \vdash h \in \mathcal{S}\!\uparrow$;
   - *if* $c = \langle h \mid E[x\, v] \rangle$, *then* $\epsilon \cup \{v\}, E :: \sigma \vdash h \in \mathcal{S}\!\uparrow$.
4. $\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow$ *implies that* $\sigma \neq \odot$ *and*
   - *if* $v \in \epsilon$, *then* $\epsilon, \sigma \vdash \langle h \mid v\, x \rangle \in \mathcal{S}\!\uparrow$ *for a fresh* x;
   - *if* $\sigma = E :: \sigma'$, *then* $\epsilon, \sigma' \vdash \langle h \mid E[x] \rangle \in \mathcal{S}\!\uparrow$ *for a fresh* x.

A normal-form simulation is a candidate relation $\mathcal{R}$ *such that* $\mathcal{R} \rightarrowtail \mathcal{R}, \mathcal{R}$, *and a bisimulation is a candidate relation* $\mathcal{R}$ *such that* $\mathcal{R}$ *and* $\mathcal{R}^{-1}$ *are simulations. Normal-form bisimilarity* $\approx$ *is the union of all normal-form bisimulations.*

When $\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d$, we reduce c until we get a value v or a stuck term $E[x\, v]$. At that point, either d also reduces to a normal form of the same kind, or we test (the first projection of) the stack Σ for divergence, assuming it is not empty. In the former case, we add the values to $\mathcal{E}$ and the evaluation contexts at the top of Σ, getting a judgment of the form $\mathcal{E}', \Sigma' \vdash h \mathrel{\mathcal{R}} g$, which then tests the environment and the stack by running either terms in $\mathcal{E}'$ or at the top of Σ'.

Example 4. We sketch the bisimulation proof for the terms t and s of Example 3. Because $\langle \emptyset \mid t \rangle \to^* \langle l := \mathsf{true} \mid f_1\, (x\ \lambda y.y) \rangle$ and $\langle \emptyset \mid s \rangle = \langle \emptyset \mid f_2\, (x\ \lambda y.y) \rangle$, we need to define $\mathcal{R}$ such that $\{(\lambda y.y, \lambda y.y)\}, (f_1\, \Box, f_2\, \Box) :: \odot \vdash l := \mathsf{true} \mathrel{\mathcal{R}} \emptyset$. Testing the equal values in the environment is easy with up-to techniques. For the contexts on the stack, we need $\{(\lambda y.y, \lambda y.y)\}, \odot \vdash \langle l := \mathsf{true} \mid f_1\ z \rangle \mathrel{\mathcal{R}} \langle \emptyset \mid f_2\ z \rangle$ for a fresh z. Since $\langle l := \mathsf{true} \mid f_1\ z \rangle \to^* \langle l := \mathsf{false} \mid \mathsf{true} \rangle$ and $\langle \emptyset \mid f_2\ z \rangle \to^* \langle \emptyset \mid \mathsf{true} \rangle$, we need $\{(\lambda y.y, \lambda y.y), (\mathsf{true}, \mathsf{true})\}, \odot \vdash l := \mathsf{false} \mathrel{\mathcal{R}} \emptyset$, which is simple to check.

Example 5. In contrast, we show that $t' \stackrel{\mathsf{def}}{=} \mathsf{new}\ l := \mathsf{true}\ \mathsf{in}\ f_1\ (x\ \lambda y.l := y; y)$ and $s' \stackrel{\mathsf{def}}{=} f_2\ (x\ \lambda y.y)$ are not bisimilar. We would need to build $\mathcal{R}$ such that $\{(\lambda y.l := y; y, \lambda y.y)\}, (f_1\, \Box, f_2\, \Box) :: \odot \vdash l := \mathsf{true} \mathrel{\mathcal{R}} \emptyset$. Testing the values in the environment, we want $\{(\lambda y.l := y; y, \lambda y.y), (z, z)\}, (f_1\, \Box, f_2\, \Box) :: \odot \vdash l := z \mathrel{\mathcal{R}} \emptyset$ for a fresh z. Executing the contexts on the stack, we get a stuck term of the form $\mathsf{if}\ z\ \mathsf{then}\ l := \mathsf{false}; \mathsf{true}\ \mathsf{else}\ \mathsf{false}$ and a value true, which cannot be related, because the former is not deferred diverging.

The terms t' and s' are therefore not bisimilar, and they are indeed not contextually equivalent, since t' gives access to its private reference by passing $\lambda y.l := y; y$ to x. The function represented by x can then change the value of l to false and break the equivalence.

$$\frac{\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d \qquad v \mathrel{\mathcal{E}} w \qquad x \notin \mathsf{fv}(v) \cup \mathsf{fv}(w)}{\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash c\{v/x\} \mathrel{\mathsf{subst_c}(\mathcal{R})} d\{w/x\}}$$

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ (E_1, F_1) :: (E_2, F_2) :: \Sigma_2 \vdash \langle h \mid t \rangle \mathrel{\mathcal{R}} \langle g \mid s \rangle}{\mathcal{E}, \Sigma_1 +\!\!+ (E_2[E_1], F_2[F_1]) :: \Sigma_2 \vdash \langle h \mid t \rangle \mathrel{\mathsf{ccomp}(\mathcal{R})} \langle g \mid s \rangle}$$

$$\frac{\mathcal{E}, (E, F) :: \Sigma \vdash \langle h \mid t \rangle \mathrel{\mathcal{R}} \langle g \mid s \rangle}{\mathcal{E}, \Sigma \vdash \langle h \mid E[t] \rangle \mathrel{\mathsf{plug}(\mathcal{R})} \langle g \mid F[s] \rangle} \qquad \frac{c \rightarrow^* c' \qquad d \rightarrow^* d' \qquad \mathcal{E}, \Sigma \vdash c' \mathrel{\mathcal{R}} d'}{\mathcal{E}, \Sigma \vdash c \mathrel{\mathsf{red}(\mathcal{R})} d}$$

$$\frac{\epsilon, \sigma \vdash \langle h \mid t \rangle \in \mathcal{R}\!\uparrow \qquad \pi_1(\mathcal{E}) = \epsilon \qquad \pi_1(\Sigma) = \sigma}{\mathcal{E}, \Sigma \vdash \langle h \mid t \rangle \mathrel{\mathsf{div}(\mathcal{R})} \langle g \mid s \rangle} \qquad \frac{\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d \qquad \mathcal{E}' \subseteq \mathcal{E}}{\mathcal{E}', \Sigma \vdash c \mathrel{\mathsf{weak}(\mathcal{R})} d}$$

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ \Sigma_2 \vdash \langle h \mid t \rangle \mathrel{\mathcal{R}} \langle g \mid s \rangle \qquad \mathsf{fr}(E) \subseteq \mathsf{dom}(h')}{\mathcal{E}, \Sigma_1 +\!\!+ (E, E) :: \Sigma_2 \vdash \langle h \uplus h' \mid t \rangle \mathrel{\mathsf{refl}(\mathcal{R})} \langle g \uplus h' \mid s \rangle}$$

Fig. 2: Selected up-to techniques for the calculus with local store

The last two cases of the bisimulation definition aim at detecting a deferred diverging context. The judgment $\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow$ roughly means that if $\sigma = E_n :: \ldots E_1 :: \odot$, then the configuration $\langle h' \mid E_1[\ldots E_n[x]] \rangle$ diverges for all fresh x and all h' obtained by running a term from $\mathcal{E}$ with the store h. As a result, when $\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow$, we have two possibilities: either we run a term from $\mathcal{E}$ in h to potentially change h, or we run the context at the top of σ (which cannot be empty in that case) to check if it is diverging. In both cases, we get a judgment of the form $\epsilon, \sigma' \vdash c \in \mathcal{R}\!\uparrow$. In that case, either c diverges and we are done, or it terminates, meaning that we have to look for divergence in σ'.

Example 6. We prove that $\langle \emptyset \mid x\, v\, \Omega \rangle$ and $\langle \emptyset \mid \Omega \rangle$ are bisimilar. We define $\mathcal{R}$ such that $\emptyset, \odot \vdash \langle \emptyset \mid x\, v\, \Omega \rangle \mathrel{\mathcal{R}} \langle \emptyset \mid \Omega \rangle$, for which we need $\{v\}, \square\, \Omega :: \odot \vdash \emptyset \in \mathcal{R}\!\uparrow$, which itself holds if $\{v\}, \odot \vdash \langle \emptyset \mid y\, \Omega \rangle \in \mathcal{R}\!\uparrow$.

Finally, only the two clauses where a reduction step takes place are active; all the others are passive, because they are simply switching from one judgment to the other without any real progress taking place. For example, when comparing value configurations, we go from a configuration judgment $\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d$ to a store judgment $\mathcal{E}, \Sigma \vdash h \mathrel{\mathcal{R}} g$ or a diverging store judgment $\mathcal{E}, \Sigma \vdash h \in \mathcal{R}\!\uparrow$. In a (diverging) store judgment, we simply decide whether we reduce a term from the store of from the stack, going back to a (diverging) configuration judgment. Actual progress is made only when we start reducing the chosen configuration.

3.3 Soundness and Completeness

We briefly discuss the up-to techniques we need to prove soundness. We write $\mathcal{E}\{(v,w)/x\}$ for the environment $\{(v'\{v/x\}, w'\{w/x\}) \mid v' \mathrel{\mathcal{E}} w'\}$, and we also define $\Sigma\{(x,w)/x\}$, $\epsilon\{v/x\}$, and $\sigma\{v/x\}$ as expected. To save space, Figure 2 presents the up-to techniques for the configuration judgment only; we give the definitions for the other judgments in Appendix A.

As in Section 2.3, the techniques subst and plug allow to reason up to substitution and plugging into an evaluation context, except that the substituted values and plugged contexts must be taken from respectively the environment and the top of the stack. The technique div relates a diverging configuration to any configuration, like in the calculus with global store. The technique ccomp allows to merge successive contexts in the stack into one. The weakening technique weak, originally known as bisimulation up to environment [16], is an usual technique for environmental bisimulations. Making the environment smaller creates a weaker judgment, as having less testing terms means a less discriminating candidate relation. Bisimulation up to reduction red is also standard and allows for a big-step reasoning by ignoring reduction steps. Finally, the technique refl allows to introduce identical contexts in the stack, but also values in the environment or terms in configurations (cf Appendix A.3).

We denote by $\mathsf{subst_c}$ the up to substitution technique restricted to the configuration and diverging configuration judgments, and by $\mathsf{subst_s}$ the restriction to the store and diverging store judgments.

Lemma 3. *The set* $\mathfrak{F} \stackrel{\mathsf{def}}{=} \{\mathsf{subst}_m, \mathsf{plug}, \mathsf{ccomp}, \mathsf{div}, \mathsf{weak}, \mathsf{red}, \mathsf{refl} \mid m \in \{\mathsf{c}, \mathsf{s}\}\}$ *is respectful, with* $\mathsf{strong}(\mathfrak{F}) = \{\mathsf{subst_s}, \mathsf{ccomp}, \mathsf{div}, \mathsf{weak}, \mathsf{red}, \mathsf{refl}\}$.

In contrast with Section 2.3 and our previous work [3], $\mathsf{subst_c}$ is *not* strong, because values are taken from the environment. Indeed, with $\mathsf{subst_c}$ strong, from $\{(v,w)\}, \odot \vdash \emptyset \mathrel{\mathcal{R}} \emptyset$, we could derive $\{(v,w)\}, \odot \vdash \langle \emptyset \mid x\, y\rangle \mathrel{\mathsf{refl}(\mathcal{R})} \langle \emptyset \mid x\, y\rangle$ and then $\{(v,w)\}, \odot \vdash \langle \emptyset \mid v\, x\rangle \mathrel{\mathsf{subst_c}(\mathsf{refl}(\mathcal{R}))} \langle \emptyset \mid w\, x\rangle$ for any v and w, which would be unsound.

The respectfulness proofs are in the appendix. Using refl, plug, $\mathsf{subst_c}$, and Lemma 1 we prove that $\approx$ is preserved by evaluation contexts and substitution, from which we deduce it is sound w.r.t. contextual equivalence.

Theorem 3. *For all t and s, if $\emptyset, \odot \vdash \langle \emptyset \mid t\rangle \approx \langle \emptyset \mid s\rangle$, then $t \equiv s$.*

To establish completeness, we follow the proof of Theorem 2, i.e., we construct a candidate relation $\mathcal{R}$ that contains $\equiv$ and prove it is a simulation by case analysis on the behavior of the related terms.

Theorem 4. *For all t and s, if $t \equiv s$, then $\emptyset, \odot \vdash \langle \emptyset \mid t\rangle \approx \langle \emptyset \mid s\rangle$.*

The main difference is that the contexts and closing substitutions are built from the environment using compatible closures [16], to take into account the private resources of the related terms. We discuss the proof in Appendix B.2.

3.4 Examples

Example 7. We start by the so-called awkward example [14,4,5]. Let

$$v \stackrel{\mathsf{def}}{=} \lambda f.l := 0; f\ (); l := 1; f\ (); !l \qquad w \stackrel{\mathsf{def}}{=} \lambda f.f\ (); f\ (); 1.$$

We equate $\mathsf{new}\ l := 0\ \mathsf{in}\ v$ and w, building the candidate $\mathcal{R}$ incrementally, starting from $\{(v, w)\}, \odot \vdash l := 0\ \mathcal{R}\ \emptyset$.

Running v and w with a fresh variable f, we obtain $\langle l := 0 \mid E_1[f\ ()]\rangle$ and $\langle \emptyset \mid E_2[f\ ()]\rangle$ with $E_1 \stackrel{\mathsf{def}}{=} \Box; l := 1; f\ (); !l$ and $F_1 \stackrel{\mathsf{def}}{=} \Box; f\ (); 1$. Ignoring the identical unit arguments (using refl), we need $\{(v, w)\}, (E_1, F_1) :: \odot \vdash l := 0\ \mathcal{R}\ \emptyset$; from that point, we can either test v and w again, resulting into an extra pair (E_1, F_1) on the stack, or run $\langle l := 0 \mid E_1[g]\rangle$ and $\langle \emptyset \mid F_1[g]\rangle$ for a fresh g instead.

In the latter case, we get $\langle l := 1 \mid E_2[g\ ()]\rangle$ and $\langle \emptyset \mid F_2[g\ ()]\rangle$, with $E_2 \stackrel{\mathsf{def}}{=} \Box; !l$ and $F_2 \stackrel{\mathsf{def}}{=} \Box; 1$, so we want $\{(v, w)\}, (E_2, F_2) :: \odot \vdash l := 1\ \mathcal{R}\ \emptyset$ (ignoring again the units). From there, testing v and w produces $\{(v, w)\}, (E_1, F_1) :: (E_2, F_2) :: \odot \vdash l := 0\ \mathcal{R}\ \emptyset$, while executing $\langle l := 1 \mid E_2[x]\rangle$ and $\langle \emptyset \mid F_2[x]\rangle$ for a fresh x gives us $\langle l := 1 \mid 1\rangle$ and $\langle \emptyset \mid 1\rangle$. This analysis suggests that $\mathcal{R}$ should be composed only of judgments of the form $\{(v, w)\}, \Sigma \vdash l := n\ \mathcal{R}\ \emptyset$ such that $n \in \{0, 1\}$ and

- Σ is an arbitrary stack composed only of pairs (E_1, F_1) or (E_2, F_2);
- if $\Sigma = (E_2, F_2) :: \Sigma'$, then $n = 1$.

We can check that such a candidate is a bisimulation, and it ensures that when l is read (when E_2 is executed), it contains the value 1.

Example 8. As a variation on the awkward example, let

$$v \stackrel{\mathsf{def}}{=} \lambda f. l := !l + 1; f\ (); l := !l - 1; !l > 0 \qquad w \stackrel{\mathsf{def}}{=} \lambda f. f\ (); \mathsf{true}.$$

We show that $\langle \emptyset \mid \mathsf{new}\ l := 1\ \mathsf{in}\ v\rangle$ and $\langle \emptyset \mid w\rangle$ are bisimilar. Let $E \stackrel{\mathsf{def}}{=} \Box; l := !l - 1; !l > 0$ and $F \stackrel{\mathsf{def}}{=} \Box; \mathsf{true}$. We write $(E, F)^n$ for the stack $\odot$ if $n = 0$ and $(E, F) :: (E, F)^{n-1}$ otherwise. Then the candidate $\mathcal{R}$ verifying $\{(v, w)\}, (E, F)^n \vdash l := n + 1\ \mathcal{R}\ \emptyset$ for any n is a bisimulation. Indeed, running v and w increases the value stored in l and adds a pair (E, F) on the stack. If $n > 0$, we can run a copy of E and F, thus decreasing the value in l by 1, and then returning true in both cases.

Example 9. This deferred divergence example comes from Dreyer et al. [4]. Let

$$v_1 \stackrel{\mathsf{def}}{=} \lambda x. \mathsf{if}\ !l\ \mathsf{then}\ \Omega\ \mathsf{else}\ k := \mathsf{true}; \lambda y.y \qquad w_1 \stackrel{\mathsf{def}}{=} \lambda x. \Omega$$

$$v_2 \stackrel{\mathsf{def}}{=} \lambda f. f\ v_1; \mathsf{if}\ !k\ \mathsf{then}\ \Omega\ \mathsf{else}\ l := \mathsf{true}; \lambda y.y \qquad w_2 \stackrel{\mathsf{def}}{=} \lambda f. f\ w_1; \lambda y.y$$

We prove that $\mathsf{new}\ l := \mathsf{false}\ \mathsf{in}\ \mathsf{new}\ k := \mathsf{false}\ \mathsf{in}\ v_2$ is equivalent to w_2. Informally, if f in w_2 applies its argument w_1, the term diverges. Divergence also happens in v_2 but in a delayed fashion, as v_1 first sets k to true, and the continuation $t \stackrel{\mathsf{def}}{=} \mathsf{if}\ !k\ \mathsf{then}\ \Omega\ \mathsf{else}\ l := \mathsf{true}; \lambda y.y$ then diverges. Similarly, if f stores w_1 or v_1 to later apply it, then divergence also occurs in both cases: in that case t sets l to true, and when v_1 is later applied, it diverges.

To build a candidate $\mathcal{R}$, we execute $\langle l := \mathsf{false}; k := \mathsf{false} \mid v_2\, f\rangle$ and $\langle \emptyset \mid w_2\, f\rangle$ for a fresh f, which gives us $\langle l := \mathsf{false}; k := \mathsf{false} \mid E[f\, v_1]\rangle$ and $\langle \emptyset \mid F[f\, w_1]\rangle$ with

$E \stackrel{\mathsf{def}}{=} \Box; t$ and $F \stackrel{\mathsf{def}}{=} \Box; \lambda y.y$. We consider $\{(v_2, w_2), (v_1, w_1)\}, (E, F) :: \emptyset \vdash l := \mathsf{false}; k := \mathsf{false}\ \mathcal{R}\ \emptyset$, for which we have several checks to do. The interesting one is running $\langle l := \mathsf{false}; k := \mathsf{false} \mid v_1\ x\rangle$ and $\langle \emptyset \mid w_1\ x\rangle$, as we get $\langle l := \mathsf{false}; k := \mathsf{true} \mid \lambda y.y\rangle$ and $\langle \emptyset \mid \Omega\rangle$. In that case, we are showing that the stack contains divergence, by establishing that $\{v_2, v_1, \lambda y.y\}, E :: \emptyset \vdash l := \mathsf{false}; k := \mathsf{true} \in \mathcal{R}{\uparrow}$, and indeed, we have $\langle l := \mathsf{false}; k := \mathsf{true} \mid E[x]\rangle \rightarrow^* \langle l := \mathsf{false}; k := \mathsf{true} \mid \Omega\rangle$ for a fresh x. In the end, the relation $\mathcal{R}$ verifying

$$
\begin{aligned}
&\{(v_2, w_2), (v_1, w_1)\}, (E, F)^n \vdash l := \mathsf{false}; k := \mathsf{false}\ \mathcal{R}\ \emptyset \\
&\{(v_2, w_2), (v_1, w_1)\}, (E, F)^n \vdash \langle l := \mathsf{false}; k := \mathsf{true} \mid \lambda y.y\rangle\ \mathcal{R}\ \langle \emptyset \mid \Omega\rangle \\
&\qquad\qquad \{v_2, v_1, \lambda y.y\}, E^n \vdash l := \mathsf{false}; k := \mathsf{true} \in \mathcal{R}{\uparrow} \\
&\qquad\qquad \{v_2, v_1, \lambda y.y\}, E^n \vdash \langle l := \mathsf{false}; k := \mathsf{true} \mid \Omega\rangle \in \mathcal{R}{\uparrow} \\
&\{(v_2, w_2), (v_1, w_1)\}, (E, F)^n \vdash l := \mathsf{true}; k := \mathsf{false}\ \mathcal{R}\ \emptyset \\
&\{(v_2, w_2), (v_1, w_1)\}, (E, F)^n \vdash \langle l := \mathsf{true}; k := \mathsf{false} \mid \Omega\rangle\ \mathcal{R}\ \langle \emptyset \mid \Omega\rangle
\end{aligned}
$$

for all n is a bisimulation up to refl and red.

4 Related Work and Conclusion

Related work. As pointed out in Section 1, the other bisimilarities defined for state either feature universal quantification over testing arguments [8,18,16,11], or are complete only for a more expressive language [17]. Kripke logical relations [1,4] also involve quantification over arguments when testing terms of a functional type. Finally, denotational models [9,12] can also be used to prove program equivalence, by showing that the denotations of two terms are equal. However, computing such denotations is difficult in general, and the automation of this task is so far restricted to a language with first-order references [13].

The work most closely related to ours is Jaber and Tabareau's Kripke Open Bisimulation (KOB) [5]. A KOB tests functional terms with fresh variables and not with related values like a regular logical relation would do. To relate two given configurations, one has to provide a World Transition System (WTS) which states the invariants the heaps of the configurations should satisfy and how to go from one invariant to the other during the evaluation. Similarly, the bisimulations for the examples of Section 3.4 state properties which could be seen as invariants about the stores at different points of the evaluation.

The difficulty for KOB as well as with our bisimilarity is to come up with the right invariants about the heaps, expressed either as a WTS or as a bisimulation. We believe that choosing a technique over the other is just a matter of preference, depending on whether one is more comfortable with game semantics or with coinduction. It would be interesting to see if there is a formal correspondence between KOB and our bisimilarity; we leave this question as a future work.

Conclusion. We define a sound and complete normal-form bisimilarity for higher-order local state, with an environment to be able to run terms in different stores.

We distinguish in the environment values which should be tested several times from the contexts which should be executed only once. The other difficulty is to relate deferred and regular diverging terms, which is taken care of by the specific judgments about divergence. The lack of quantification over arguments make the bisimulation proofs quite simple.

A future work would be to make these proofs even simpler by defining appropriate up-to techniques. The techniques we use in Section 3.3 to prove soundness turn out to be not that useful when establishing the equivalences of Section 3.4, except for trivial ones such as up to reduction or reflexivity. The difficulty in defining the candidate relations for the examples of Section 3.4 is in finding the right property relating the stack Σ to the store, so maybe an up-to technique could make this task easier.

As pointed out in Section 1, our results can be seen as an indication of what kind of additional infrastructure in a complete normal-form bisimilarity is required when the considered syntactic theory becomes less discriminative—in our case, when control operators vanish from the picture, and mutable state is the only extension of the λ-calculus. A question one could then ask is whether we can find a less expressive calculus—maybe the plain λ-calculus itself—for which a suitably enhanced normal-form bisimilarity is still complete.

Acknowledgements. We thank Guilhem Jaber and the anonymous reviewers for their comments. This work was supported by National Science Centre, Poland, grant no. 2014/15/B/ST6/00619 and by COST Action EUTypes CA15123.

References

1. A. Ahmed, D. Dreyer, and A. Rossberg. State-dependent representation independence. In B. C. Pierce, editor, *Proceedings of the Thirty-Fifth Annual ACM Symposium on Principles of Programming Languages*, pages 340–353. ACM Press, Jan. 2009.
2. A. Aristizábal, D. Biernacki, S. Lenglet, and P. Polesiuk. Environmental Bisimulations for Delimited-Control Operators with Dynamic Prompt Generation. *Logical Methods in Computer Science*, 13(3), 2017.
3. D. Biernacki, S. Lenglet, and P. Polesiuk. Proving soundness of extensional normal-form bisimilarities. In A. Silva, editor, *Proceedings of the 33th Annual Conference on Mathematical Foundations of Programming Semantics(MFPS XXXIII)*, volume 336 of *Electronic Notes in Theoretical Computer Science*, pages 41–56, Ljubljana, Slovenia, June 2017.
4. D. Dreyer, G. Neis, and L. Birkedal. The impact of higher-order state and control effects on local relational reasoning. *Journal of Functional Programming*, 22(4-5):477–528, 2012.
5. G. Jaber and N. Tabareau. Kripke open bisimulation – A marriage of game semantics and operational techniques. In X. Feng and S. Park, editors, *Programming Languages and Systems – 13th Asian Symposium, APLAS 2015*, volume 9458 of *Lecture Notes in Computer Science*, pages 271–291, Pohang, South Korea, Nov. 2015. Springer.
6. R. Jagadeesan, C. Pitcher, and J. Riely. Open bisimulation for aspects. *Trans. Aspect-Oriented Software Development*, 5:72–132, 2009.
7. V. Koutavas, P. B. Levy, and E. Sumii. From applicative to environmental bisimulation. In M. Mislove and J. Ouaknine, editors, *Proceedings of the 27th Annual Conference on Mathematical Foundations of Programming Semantics(MFPS XXVII)*, volume 276 of *ENTCS*, pages 215–235, Pittsburgh, PA, USA, May 2011.
8. V. Koutavas and M. Wand. Small bisimulations for reasoning about higher-order imperative programs. In J. G. Morrisett and S. L. P. Jones, editors, *POPL'06*, pages 141–152, Charleston, SC, USA, Jan. 2006. ACM Press.
9. J. Laird. A fully abstract trace semantics for general references. In L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, editors, *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, 2007, Proceedings*, volume 4596 of *Lecture Notes in Computer Science*, pages 667–679, Wroclaw, Poland, July 2007. Springer.
10. S. B. Lassen. Eager normal form bisimulation. In P. Panangaden, editor, *LICS'05*, pages 345–354, Chicago, IL, June 2005. IEEE Computer Society Press.
11. J. Madiot, D. Pous, and D. Sangiorgi. Bisimulations up-to: Beyond first-order transition systems. In P. Baldan and D. Gorla, editors, *25th International Conference on Concurrency Theory*, volume 8704 of *Lecture Notes in Computer Science*, pages 93–108, Rome, Italy, Sept. 2014. Springer.
12. A. S. Murawski and N. Tzevelekos. Game semantics for good general references. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011*, pages 75–84. IEEE Computer Society, June 2011.
13. A. S. Murawski and N. Tzevelekos. Algorithmic games for full ground references. *Formal Methods in System Design*, 52(3):277–314, 2018.
14. A. Pitts and I. Stark. Operational reasoning for functions with local state. In A. Gordon and A. Pitts, editors, *Higher Order Operational Techniques in Semantics*, pages 227–273. Publications of the Newton Institute, Cambridge University Press, 1998.

15. D. Sangiorgi. The lazy lambda calculus in a concurrency scenario. In A. Scedrov, editor, *LICS'92*, pages 102–109, Santa Cruz, California, June 1992. IEEE Computer Society.
16. D. Sangiorgi, N. Kobayashi, and E. Sumii. Environmental bisimulations for higher-order languages. *ACM Transactions on Programming Languages and Systems*, 33(1):1–69, Jan. 2011.
17. K. Støvring and S. B. Lassen. A complete, co-inductive syntactic theory of sequential control and state. In M. Felleisen, editor, *POPL'07*, SIGPLAN Notices, Vol. 42, No. 1, pages 161–172, Nice, France, Jan. 2007. ACM Press.
18. E. Sumii. A complete characterization of observational equivalence in polymorphic *lambda*-calculus with general references. In E. Grädel and R. Kahle, editors, *Computer Science Logic, 23rd international Workshop, CSL 2009, 18th Annual Conference of the EACSL, Proceedings*, volume 5771 of *Lecture Notes in Computer Science*, pages 455–469, Coimbra, Portugal, Sept. 2009. Springer.

A Respectfulness Proofs

For the proofs, we write $f \rightsquigarrow$ (respectively $f \rightsquigarrow_{\mathsf{s}}$) if there exist f' and f'' such that $f \rightsquigarrow f', f''$ (respectively $f \rightsquigarrow_{\mathsf{s}} f', f''$) and the conditions of Definition 4 are met.

A.1 Up to substitution

$$\frac{\mathcal{E}, \Sigma \vdash c \mathrel{\mathcal{R}} d \qquad v \mathrel{\mathcal{E}} w \qquad x \notin \mathsf{fv}(v) \cup \mathsf{fv}(w)}{\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash c\{v/x\} \mathrel{\mathsf{subst_c}(\mathcal{R})} d\{w/x\}}$$

$$\frac{\mathcal{E}, \Sigma \vdash h \mathrel{\mathcal{R}} g \qquad v \mathrel{\mathcal{E}} w \qquad x \notin \mathsf{fv}(v) \cup \mathsf{fv}(w)}{\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash h\{v/x\} \mathrel{\mathsf{subst_s}(\mathcal{R})} g\{w/x\}}$$

$$\frac{\epsilon, \sigma \vdash h \in \mathcal{R}{\uparrow} \qquad v \in \epsilon \qquad x \notin \mathsf{fv}(v)}{\epsilon\{v/x\}, \sigma\{v/x\} \vdash h\{v/x\} \in \mathsf{subst_{s\uparrow}}(\mathcal{R}){\uparrow}}$$

$$\frac{\epsilon, \sigma \vdash c \in \mathcal{R}{\uparrow} \qquad v \in \epsilon \qquad x \notin \mathsf{fv}(v)}{\epsilon\{v/x\}, \sigma\{v/x\} \vdash c\{v/x\} \in \mathsf{subst_{\uparrow}}(\mathcal{R}){\uparrow}}$$

Lemma 4. $\mathsf{subst_c} \rightsquigarrow$

Proof. Let $\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash c\{v/x\} \mathrel{\mathsf{subst_c}(\mathcal{R})} d\{w/x\}$ with $c = \langle h \mid t\rangle$ and $d = \langle g \mid s\rangle$. We proceed by case analysis on t.

If $\langle h \mid t\rangle \to \langle h' \mid t'\rangle$, then there exist g', s' such that $\langle g \mid s\rangle \to^* \langle g' \mid s'\rangle$ and $\mathcal{E}, \Sigma \vdash \langle h' \mid t'\rangle \mathrel{\mathcal{S}} \langle g' \mid s'\rangle$. Hence we have $\langle g\{w/x\} \mid s\{w/x\}\rangle \to^* \langle g'\{w/x\} \mid s'\{w/x\}\rangle$ with $\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash \langle h'\{v/x\} \mid t'\{v/x\}\rangle \mathrel{\mathsf{subst_c}(\mathcal{S})} \langle g'\{w/x\} \mid s\{w/x\}\rangle$, as wished.

If $t = v'$, then there exist g_1', w' such that $\langle g_1 \mid s\rangle \to^* \langle g_1' \mid w'\rangle$ and $\mathcal{E} \cup \{(v', w')\}, \Sigma \vdash h_1 \mathrel{\mathcal{R}} g_1'$. We conclude with $\mathsf{subst_s}$.

Suppose $t = E[zv']$ and there exist g', w', F such that $\langle g \mid s\rangle \to^* \langle g' \mid F[zw']\rangle$ and $\mathcal{E} \cup \{(v',w')\}, (E,F) :: \Sigma \vdash h \mathrel{\mathcal{R}} g'$. If $z \neq x$, then we conclude using $\mathsf{subst_s}$. Suppose $z = x$. Then $\mathcal{E} \cup \{(v',w')\}, (E,F) :: \Sigma \vdash h \mathrel{\mathcal{R}} g'$ implies $\mathcal{E} \cup \{(v',w')\}, (E,F) :: \Sigma \vdash \langle h \mid v\, y\rangle \mathrel{\mathcal{R}} \langle g' \mid w\, y\rangle$ for a fresh y. We distinguish two cases. If $\langle h \mid vy\rangle \to \langle h' \mid t'\rangle$, then there exist g'', s' such that $\langle g' \mid wy\rangle \to^* \langle g'' \mid s'\rangle$ and $\mathcal{E} \cup \{(v'\, z', w'\, z')\}, (E,F) :: \Sigma \vdash \langle h' \mid t'\rangle \mathrel{\mathcal{S}} \langle g'' \mid s'\rangle$. As a result, we have

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\} \cup \{(v'\{v/x\}, w'\{w/x\})\}, (E\{v/x\}, F\{w/x\}) :: \Sigma\{(v,w)/x\} \vdash \\ &\qquad\qquad\qquad \langle h'\{v/x\} \mid t'\rangle \mathrel{\mathsf{subst_c}(\mathcal{S})} \langle g''\{v/x\} \mid s'\rangle;\end{aligned}$$

because $x \notin \mathsf{fv}(v\, y) \cup \mathsf{fv}(w\, y)$, we know that $x \notin \mathsf{fv}(t') \cup \mathsf{fv}(s')$. We then deduce that

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\} \cup \{(v'\{v/x\}, w'\{w/x\})\}, (E\{v/x\}, F\{w/x\}) :: \Sigma\{(v,w)/x\} \vdash \\ &\quad \langle h'\{v/x\} \mid t'\{v'\{v/x\}/y\}\rangle \mathrel{\mathsf{subst_c}(\mathsf{subst_c}(\mathcal{S}))} \langle g''\{v/x\} \mid s'\{w'\{w/x\}/y\}\rangle;\end{aligned}$$

again, because y is fresh except w.r.t. t' and s', the substitution occurs in these two terms only. We then obtain

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash \langle h'\{v/x\} \mid E\{v/x\}[t'\{v'\{v/x\}/y\}]\rangle \\ &\qquad\qquad \mathsf{weak}(\mathsf{plug_c}(\mathsf{subst_c}(\mathsf{subst_c}(\mathcal{S})))) \ \langle g''\{v/x\} \mid F\{w/x\}[s'\{w'\{w/x\}/y\}]\rangle\end{aligned}$$

and we can check that $\langle h\{v/x\} \mid t\{v/x\}\rangle \to \langle h'\{v/x\} \mid E\{v/x\}[t'\{v'\{v/x\}/y\}]\rangle$ and $\langle g\{w/x\} \mid s\{w/x\}\rangle \to^* \langle g''\{w/x\} \mid F\{w/x\}[s'\{w'\{w/x\}/y\}]\rangle$, so the result holds.

Otherwise, $v\ y$ is an open-stuck term, i.e., $v = x'$ for some x'. There exists F', g'', and w'' such that $\langle g' \mid w\ y\rangle \to^* \langle g'' \mid F'[x'\ w'']\rangle$ and $\mathcal{E} \cup \{(v', w')\} \cup \{(y, w'')\}, (\Box, F') :: (E, F) :: \Sigma \vdash h\ \mathcal{R}\ g''$. It implies

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\} \cup \{(v'\{v/x\}, w'\{v/x\})\} \cup \{(y, w'')\}, \\ &\quad (\Box, F') :: (E\{v/x\}, F\{w/x\}) :: \Sigma\{(v,w)/x\} \vdash h\{v/x\}\ \mathsf{subst_s}(\mathcal{R})\ g''\{w/x\};\end{aligned}$$

again we know that x does not occur in w'' and F' because it does not occur in v and w. Then we get

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\} \cup \{(v'\{v/x\}, w'\{w/x\})\} \cup \{(v'\{v/x\}, w''\{w'\{w/x\}/y\})\}, \\ &(\Box, F') :: (E\{v/x\}, F\{w/x\}) :: \Sigma\{(v,w)/x\} \vdash h\{v/x\}\ \mathsf{subst_s}(\mathsf{subst_s}(\mathcal{R}))\ g''\{w/x\}\end{aligned}$$

and finally

$$\begin{aligned}&\mathcal{E}\{(v,w)/x\} \cup \{(v'\{v/x\}, w''\{w'\{w/x\}/y\})\}, \\ &\qquad\qquad (E\{v/x\}, F\{w/x\}[F']) :: \Sigma\{(v,w)/x\} \vdash h\{v/x\} \\ &\qquad\qquad\qquad\qquad \mathsf{weak}(\mathsf{ccomp}(\mathsf{subst_s}(\mathsf{subst_s}(\mathcal{R})))) \ g''\{w/x\}\end{aligned}$$

which is what we want, because $\langle h\{v/x\} \mid t\{v/x\}\rangle = \langle h\{v/x\} \mid E\{v/x\}[x' v'\{v/x\}]\rangle$, and $\langle g\{w/x\} \mid s\{w/x\}\rangle \to^* \langle g''\{w/x\} \mid F\{w/x\}[F'[x'\ w''\{w'\{w/x\}/y\}]]\rangle$, and all the up-to techniques used are strong (even $\mathsf{subst_s}$).

Suppose $t = E[z\ v']$ and $\pi_1(\mathcal{E}) \cup \{v'\}, E :: \pi_1(\Sigma) \vdash h \in \mathcal{R}{\uparrow}$. If $z \neq x$ or if $z = x$ and $v = x'$ for some x', then we conclude with $\mathsf{subst_{s\uparrow}}$. Otherwise, we have $\pi_1(\mathcal{E}) \cup \{v'\}, E :: \pi_1(\Sigma) \vdash \langle h \mid v\,y\rangle \in \mathcal{R}{\uparrow}$ for a fresh y, and $\langle h \mid v\,y\rangle \to \langle h' \mid t'\rangle$ for some t' and h'. It implies $\pi_1(\mathcal{E}) \cup \{v'\}, E :: \pi_1(\Sigma) \vdash \langle h' \mid t'\rangle \in \mathcal{S}{\uparrow}$, from which we get

$$\pi_1(\mathcal{E}) \cup \{v'\}, \pi_1(\Sigma) \vdash \langle h' \mid E\{v/x\}[t'\{v'\{v/x\}/y\}]\rangle \in \mathsf{plug}_{\uparrow}(\mathsf{subst}_{\uparrow}(\mathsf{subst}_{\uparrow}(\mathcal{S}))){\uparrow}.$$

As a result, we have

$$\mathcal{E}, \Sigma \vdash \langle h' \mid E\{v/x\}[t'\{v'\{v/x\}/y\}]\rangle\ \mathsf{div}(\mathsf{weak}(\mathsf{plug}_{\uparrow}(\mathsf{subst}_{\uparrow}(\mathsf{subst}_{\uparrow}(\mathcal{S}))))) \ \langle g \mid s\rangle,$$

as wished.

Lemma 5. $\mathsf{subst_s} \rightsquigarrow_{\mathsf{s}}$

Proof. From $\mathcal{E}, \Sigma \vdash h \mathrel{\mathcal{R}} g$, we get $\mathcal{E}, \Sigma \vdash \langle h \mid v\, y\rangle \mathrel{\mathcal{S}} \langle h \mid v\, y\rangle$ for a fresh y, which with $v' \mathrel{\mathcal{E}} w'$ implies $\mathcal{E}\{(v,w)/x\}, \Sigma\{(v,w)/x\} \vdash \langle h\{v/x\} \mid v'\{v/x\}\, y\rangle \mathrel{\mathsf{subst_c}(\mathcal{S})} \langle h\{w/x\} \mid w'\{w/x\}\, y\rangle$, as wished. Similarly, from $\mathcal{E}, (E, F) :: \Sigma' \vdash h \mathrel{\mathcal{R}} g$, we get $\mathcal{E}, \Sigma' \vdash \langle h \mid E[y]\rangle \mathrel{\mathcal{S}} \langle g \mid F[y]\rangle$ for a fresh y, which in turn implies

$$\mathcal{E}\{(v,w)/x\}, \Sigma'\{(v,w)/x\} \vdash \langle h\{v/x\} \mid E\{v/x\}[y]\rangle \mathrel{\mathsf{subst_c}(\mathcal{S})} \langle g\{w/x\} \mid F\{w/x\}[y]\rangle.$$

Lemma 6. $\mathsf{subst_{s\uparrow}} \rightsquigarrow_{\mathsf{s}} \mathsf{subst_{\uparrow}}, \mathsf{id}$

Proof. Same as the previous one.

Lemma 7. $\mathsf{subst_{\uparrow}} \rightsquigarrow$

Proof. If $\langle h \mid t\rangle \to \langle h' \mid t'\rangle$, then $\epsilon, \sigma \vdash \langle h' \mid t'\rangle \in \mathcal{S}{\uparrow}$, which implies $\epsilon\{v/x\}, \sigma\{v/x\} \vdash \langle h'\{v/x\} \mid t'\{v/x\}\rangle \in \mathsf{subst_{\uparrow}}(\mathcal{S}){\uparrow}$, as wished. The case $t = v'$ is similar.

If $t = E[z\, v']$, then $\epsilon \cup \{v'\}, E :: \sigma \vdash h \in \mathcal{R}{\uparrow}$. The case $z \neq x$ is easy. Suppose $z = x$. If $v = x'$ for some x', then $\epsilon\{x'/x\} \cup \{v'\}, E\{x'/x\} :: \sigma\{x'/x\} \vdash h\{x'/x\} \in \mathsf{subst_{s\uparrow}}(\mathcal{R}){\uparrow}$ holds, as wished.

Otherwise, because $v \in \epsilon$, we have $\epsilon \cup \{v'\}, E :: \sigma \vdash \langle h \mid v\, y\rangle \in \mathcal{R}{\uparrow}$ for some y. But we also have $\langle h \mid v y\rangle \to \langle h' \mid t'\rangle$ for some h', and t', therefore $\epsilon \cup \{v'\}, E :: \sigma \vdash \langle h' \mid t'\rangle \in \mathcal{S}{\uparrow}$ holds, which implies $\epsilon \cup \{v'\}, E :: \sigma \vdash \langle h' \mid t'\{v'/y\}\rangle \in \mathsf{subst_{\uparrow}}(\mathcal{S}){\uparrow}$, and then

$$\begin{aligned}&\epsilon\{v/x\}, \sigma\{v/x\} \vdash \\ &\qquad \langle h'\{v/x\} \mid E\{v/x\}[t'\{v'\{v/x\}/y\}]\rangle \in \mathsf{weak}(\mathsf{plug_{\uparrow}}(\mathsf{subst_{\uparrow}}(\mathsf{subst_{\uparrow}}(\mathcal{S})))){\uparrow}.\end{aligned}$$

A.2 Up to Context Plugging and Composition

$$\frac{\mathcal{E}, (E, F) :: \Sigma \vdash \langle h \mid t\rangle \mathrel{\mathcal{R}} \langle g \mid s\rangle}{\mathcal{E}, \Sigma \vdash \langle h \mid E[t]\rangle \mathrel{\mathsf{plug_c}(\mathcal{R})} \langle g \mid F[s]\rangle} \qquad \frac{\epsilon, E :: \sigma \vdash \langle h \mid t\rangle \in \mathcal{R}{\uparrow} \qquad \sigma \neq \odot}{\epsilon, \sigma \vdash \langle h \mid E[t]\rangle \in \mathsf{plug_{\uparrow}}(\mathcal{R}){\uparrow}}$$

Lemma 8. $\mathsf{plug_c} \rightsquigarrow$

Proof. By case analysis on t. If $\langle h \mid t\rangle \to \langle h' \mid t'\rangle$, then there exist g', s' such that $\langle g \mid s\rangle \to^* \langle g' \mid s'\rangle$ and $\mathcal{E}, \Sigma \vdash \langle h' \mid t'\rangle \mathrel{\mathcal{S}} \langle g' \mid s'\rangle$. We conclude with $\mathsf{plug_c}$.

If $t = v$, then $\langle g \mid s\rangle \to^* \langle g' \mid w\rangle$ and $\mathcal{E} \cup \{(v,w)\}, (E,F) :: \Sigma \vdash h \mathrel{\mathcal{R}} g'$ for some g' and w. Hence, we have $\mathcal{E} \cup \{(v,w)\}, \Sigma \vdash \langle h \mid E[z]\rangle \mathrel{\mathcal{R}} \langle g' \mid F[z]\rangle$ for a fresh z, which implies $\mathcal{E}, \Sigma \vdash \langle h \mid E[v]\rangle \mathrel{\mathsf{weak}(\mathsf{subst_c}(\mathcal{R}))} \langle g' \mid F[w]\rangle$. From there, we conclude as with $\mathsf{subst_c}$.

Suppose $t = E'[x\, v]$ and there exist g', w, and F' such that $\langle g \mid s\rangle \to^* \langle g' \mid F'[x\, w]\rangle$ and $\mathcal{E} \cup \{(v,w)\}, (E', F') :: (E, F) :: \Sigma \vdash h \mathrel{\mathcal{R}} g'$. Then $E[t]$ is also open stuck, $\langle g \mid F[s]\rangle \to^* \langle g' \mid F[F'[xw]]\rangle$, and we have $\mathcal{E} \cup \{(v,w)\}, (E[E'], F[F']) :: \Sigma \vdash h \mathrel{\mathsf{ccomp}(\mathcal{R})} g'$, as wished.

If $t = E'[xv]$ and $\pi_1(\mathcal{E}), E' :: E :: \pi_1(\Sigma) \vdash h \in \mathcal{R}{\uparrow}$; then $\pi_1(\mathcal{E}), E[E'] :: \pi_1(\Sigma) \vdash h \in \mathsf{ccomp}(\mathcal{R}){\uparrow}$.

Lemma 9. $\mathsf{plug_{\uparrow}} \rightsquigarrow$

Proof. By case analysis on t. If $\langle h \mid t\rangle \to \langle h' \mid t'\rangle$, then $\epsilon, E :: \sigma \vdash \langle h' \mid t'\rangle \in \mathcal{S}{\uparrow}$, and we conclude with $\mathsf{plug}_{\uparrow}$.

If $t = v$, then $\epsilon \cup \{v\}, E :: \sigma \vdash h \in \mathcal{R}{\uparrow}$ for a fresh x. Then $\epsilon \cup \{v\}, \sigma \vdash \langle h \mid E[y]\rangle \in \mathcal{R}{\uparrow}$, which implies $\epsilon, \sigma \vdash \langle h \mid E[v]\rangle \in \mathsf{weak}(\mathsf{subst}_{\uparrow}(\mathcal{R})){\uparrow}$, so we conclude as with $\mathsf{subst}_{\uparrow}$.

If $t = E'[x\ v]$ then $\epsilon \cup \{v\}, E' :: E :: \sigma \vdash h \in \mathcal{R}{\uparrow}$ for a fresh x, and $\epsilon \cup \{v\}, E[E'] :: \sigma \vdash h \in \mathsf{ccomp}(\mathcal{R}){\uparrow}$.

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ (E_1, F_1) :: (E_2, F_2) :: \Sigma_2 \vdash h\ \mathcal{R}\ g}{\mathcal{E}, \Sigma_1 +\!\!+ (E_2[E_1], F_2[F_1]) :: \Sigma_2 \vdash h\ \mathsf{ccomp}(\mathcal{R})\ g}$$

$$\frac{\epsilon, \sigma_1 +\!\!+ E_1 :: E_2 :: \sigma_2 \vdash h \in \mathcal{R}{\uparrow}}{\epsilon, \sigma_1 +\!\!+ E_2[E_1] :: \sigma_2 \vdash h \in \mathsf{ccomp}(\mathcal{R}){\uparrow}}$$

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ (E_1, F_1) :: (E_2, F_2) :: \Sigma_2 \vdash \langle h \mid t\rangle\ \mathcal{R}\ \langle g \mid s\rangle}{\mathcal{E}, \Sigma_1 +\!\!+ (E_2[E_1], F_2[F_1]) :: \Sigma_2 \vdash \langle h \mid t\rangle\ \mathsf{ccomp}(\mathcal{R})\ \langle g \mid s\rangle}$$

$$\frac{\epsilon, \sigma_1 +\!\!+ E_1 :: E_2 :: \sigma_2 \vdash \langle h \mid t\rangle \in \mathcal{R}{\uparrow}}{\epsilon, \sigma_1 +\!\!+ E_2[E_1] :: \sigma_2 \vdash \langle h \mid t\rangle \in \mathsf{ccomp}(\mathcal{R}){\uparrow}}$$

Lemma 10. $\mathsf{ccomp} \rightsquigarrow_{\mathsf{s}}$

Proof. The clause for checking terms of the environment is easy using ccomp. For the other check, if Σ_1 is not empty, we can conclude with ccomp. If Σ_1 is empty, then from $\mathcal{E}, (E_1, F_1) :: (E_2, F_2) :: \Sigma_2 \vdash h\ \mathcal{R}\ g$, we get $\mathcal{E}, (E_2, F_2) :: \Sigma_2 \vdash \langle h \mid E_1[x]\rangle\ \mathcal{S}\ \langle g \mid F_1[x]\rangle$ for a fresh x, and then $\mathcal{E}, \Sigma_2 \vdash \langle h \mid E_2[E_1[x]]\rangle\ \mathsf{plug}_{\mathsf{c}}(\mathcal{S})\ \langle g \mid F_2[F_1[x]]\rangle$.

Lemma 11. $\mathsf{ccomp} \rightsquigarrow_{\mathsf{s}}$

Proof. Same as the previous one.

Lemma 12. $\mathsf{ccomp} \rightsquigarrow_{\mathsf{s}}$

Proof. By case analysis on t. In all the cases, we are just unfolding the definitions.

Lemma 13. $\mathsf{ccomp} \rightsquigarrow_{\mathsf{s}}$

Proof. Same as the previous one.

A.3 Reflexivity

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ \Sigma_2 \vdash \langle h \mid t\rangle \; \mathcal{R} \; \langle g \mid s\rangle \qquad \mathsf{fr}(E) \subseteq \mathsf{dom}(h')}{\mathcal{E}, \Sigma_1 +\!\!+ (E, E) :: \Sigma_2 \vdash \langle h \uplus h' \mid t\rangle \; \mathsf{refl}(\mathcal{R}) \; \langle g \uplus h' \mid t\rangle}$$

$$\frac{\mathcal{E}, \Sigma_1 +\!\!+ \Sigma_2 \vdash h \; \mathcal{R} \; g \qquad \mathsf{fr}(E) \subseteq \mathsf{dom}(h')}{\mathcal{E}, \Sigma_1 +\!\!+ (E, E) :: \Sigma_2 \vdash h \uplus h' \; \mathsf{refl}(\mathcal{R}) \; g \uplus h'}$$

$$\frac{\epsilon, \sigma_1 +\!\!+ \sigma_2 \vdash \langle h \mid t\rangle \in \mathcal{R}\!\uparrow \qquad \mathsf{fr}(E) \subseteq \mathsf{dom}(h')}{\epsilon, \sigma_1 +\!\!+ E :: \sigma_2 \vdash \langle h \uplus h' \mid t\rangle \in \mathsf{refl}(\mathcal{R})\!\uparrow}$$

$$\frac{\epsilon, \sigma_1 +\!\!+ \sigma_2 \vdash h \in \mathcal{R}\!\uparrow \qquad \mathsf{fr}(E) \subseteq \mathsf{dom}(h')}{\epsilon, \sigma_1 +\!\!+ E :: \sigma_2 \vdash h \uplus h' \in \mathsf{refl}(\mathcal{R})\!\uparrow}$$

$$\frac{\mathcal{E}, \Sigma \vdash h \; \mathcal{R} \; g \qquad \mathsf{fr}(t) \subseteq \mathsf{dom}(h')}{\mathcal{E}, \Sigma \vdash \langle h \uplus h' \mid t\rangle \; \mathsf{refl}(\mathcal{R}) \; \langle g \uplus h' \mid t\rangle} \qquad \frac{\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow \qquad \mathsf{fr}(t) \subseteq \mathsf{dom}(h')}{\epsilon, \sigma \vdash \langle h \uplus h' \mid t\rangle \in \mathsf{refl}(\mathcal{R})\!\uparrow}$$

$$\frac{\mathcal{E}, \Sigma \vdash h \; \mathcal{R} \; g \qquad \mathsf{fr}(v) \subseteq \mathsf{dom}(h')}{\mathcal{E} \cup \{(v, v)\}, \Sigma \vdash h \uplus h' \; \mathsf{refl}(\mathcal{R}) \; g \uplus h'} \qquad \frac{\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow \qquad \mathsf{fr}(v) \subseteq \mathsf{dom}(h')}{\epsilon \cup \{v\}, \sigma \vdash h \uplus h' \in \mathsf{refl}(\mathcal{R})\!\uparrow}$$

$$\frac{\mathcal{E}, \Sigma \vdash \langle h \mid t\rangle \; \mathcal{R} \; \langle g \mid s\rangle \qquad \mathsf{fr}(v) \subseteq \mathsf{dom}(h')}{\mathcal{E} \cup \{(v, v)\}, \Sigma \vdash \langle h \uplus h' \mid t\rangle \; \mathsf{refl}(\mathcal{R}) \; \langle g \uplus h' \mid s\rangle}$$

$$\frac{\epsilon, \sigma \vdash \langle h \mid t\rangle \in \mathcal{R}\!\uparrow \qquad \mathsf{fr}(v) \subseteq \mathsf{dom}(h')}{\epsilon \cup \{v\}, \sigma \vdash \langle h \uplus h' \mid t\rangle \in \mathsf{refl}(\mathcal{R})\!\uparrow}$$

Lemma 14. $\mathsf{refl} \rightsquigarrow_{\mathsf{s}}$

Proof. The context E or value v may allocate fresh references, hence the need for h'. Other than that, the proofs are straightforward.

A.4 Other Techniques

$$\frac{\epsilon, \sigma \vdash \langle h \mid t\rangle \in \mathcal{R}\!\uparrow \qquad \pi_1(\mathcal{E}) = \epsilon \qquad \pi_1(\Sigma) = \sigma}{\mathcal{E}, \Sigma \vdash \langle h \mid t\rangle \; \mathsf{div}(\mathcal{R}) \; \langle g \mid s\rangle}$$

Lemma 15. $\mathsf{div} \rightsquigarrow_{\mathsf{s}}$

Proof. By case analysis on t; each case is straightforward.

$$\frac{\mathcal{E}, \Sigma \vdash c \; \mathcal{R} \; d \qquad \mathcal{E}' \subseteq \mathcal{E}}{\mathcal{E}', \Sigma \vdash c \; \mathsf{weak}(\mathcal{R}) \; d} \qquad \frac{\mathcal{E}, \Sigma \vdash h \; \mathcal{R} \; g \qquad \mathcal{E}' \subseteq \mathcal{E}}{\mathcal{E}', \Sigma \vdash c \; \mathsf{weak}(\mathcal{R}) \; d}$$

$$\frac{\epsilon, \sigma \vdash c \in \mathcal{R}\!\uparrow \qquad \epsilon' \subseteq \epsilon}{\epsilon', \Sigma \vdash c \in \mathsf{weak}(\mathcal{R})\!\uparrow} \qquad \frac{\epsilon, \sigma \vdash h \in \mathcal{R}\!\uparrow \qquad \epsilon' \subseteq \epsilon}{\epsilon', \Sigma \vdash c \in \mathsf{weak}(\mathcal{R})\!\uparrow}$$

Lemma 16. weak $\rightsquigarrow_{\mathsf{s}}$

Proof. Straightforward

$$\frac{c \to^* c' \qquad d \to^* d' \qquad \mathcal{E}, \Sigma \vdash c' \mathrel{\mathcal{R}} d'}{\mathcal{E}, \Sigma \vdash c \mathrel{\mathsf{red}(\mathcal{R})} d}$$

Lemma 17. red $\rightsquigarrow_{\mathsf{s}}$

Proof. Straightforward

B Completeness Proofs

B.1 Proof of Theorem 2

Given a store h such that $\mathsf{dom}(h) = \{l_1 \dots l_n\}$ and a term t, we write $\bigcup_i l_i := h; t$ for the term $l_1 := h(l_1); \dots l_n := h(l_n); t$. In the following we implicitly take advantage of the fact that if $h = h' \uplus h''$, then $\langle h \mid t \rangle \Downarrow$ iff $\langle h' \mid t \rangle \Downarrow$.

Lemma 18. *The relation $\mathcal{R}$ defined by*

$$\begin{aligned}&\{(\langle h \mid t \rangle, \langle g \mid s \rangle) \mid \forall E, h_E, \ \textit{closing } \int, \langle h \uplus h_E \mid E[t] \rangle \int \Downarrow \ \Rightarrow \langle g \uplus h_E \mid E[s] \rangle \int \Downarrow\} \\ &\cup \{\langle h \mid t \rangle \mid \forall E, h_E, \ \textit{closing } \int, \langle h \uplus h_E \mid E[t] \rangle \int \Uparrow\}\end{aligned}$$

is a simulation.

Proof. Let $\langle h \mid t \rangle \mathrel{\mathcal{R}} \langle g \mid s \rangle$. We consider the possible cases in turn.

Case $\langle h \mid t \rangle \to c'$. Follows immediately from the definition of $\mathcal{R}$.

Case $t = v$. We have to check first that $\langle g \mid s \rangle$ evaluates to a value configuration. By definition of $\mathcal{R}$, since $\langle h \mid v \rangle$ is a value configuration, $\langle g \mid s \rangle$ cannot diverge. Furthermore, it cannot reduce to a stuck configuration either. For if $\langle g \mid s \rangle \to^* \langle g' \mid E[x\ w] \rangle$ for some g', E, x, and w, then taking $\int$ to be the substitution which maps all the free variables of h, t, g, and s to $\lambda x.\Omega$, we get $\langle g \mid s \rangle \int \to^* \langle g' \int \mid E \int [(\lambda x.\Omega)\, w \int] \rangle$, a diverging configuration, hence a contradiction with the fact that $\langle h \mid v \rangle \int$ is a value configuration. Therefore, there exist g', w such that $\langle g \mid s \rangle \to^* \langle g' \mid w \rangle$. We have to show that $h \mathrel{\mathcal{R}^{\mathsf{h}}} g'$ and $v \mathrel{\mathcal{R}^{\mathsf{v}}} w$. For the latter, we have to show that for a fresh store h_f, a fresh variable x, a context E, a store h_E, and a closing substitution $\int$, it is the case that $\langle h_f \uplus h_E \mid E[v\ x] \rangle \int \Downarrow$ iff $\langle h_f \uplus h_E \mid E[w\ x] \rangle \int \Downarrow$. We use the fact that $\langle h \mid t \rangle \mathrel{\mathcal{R}} \langle g \mid s \rangle$ with the same $\int$, h_E, and the context $E' \stackrel{\mathsf{def}}{=} (\lambda y. \bigcup_i l_i := h_f; E[y\ x])\, \Box$; one can check that $\langle h \uplus h_E \mid E'[t] \rangle \int \to^* \langle h_f \uplus h_E \mid E[v\ x] \rangle \int$, and similarly for $\langle g \mid s \rangle$. To show $h \mathrel{\mathcal{R}^{\mathsf{h}}} g'$, we proceed similarly using $(\lambda y. \mathsf{let}\ z = !l_j\ \mathsf{in}\ \bigcup_i l_i := h_f \uplus h_E; E[z\ x])\, \Box$ for each j.

Case $t = E[x\,v]$. Suppose there exist $\int$, $h_{E'}$, and E' such that $\langle h \uplus h_{E'} \mid E'[t]\rangle\int$ terminates. We show that $\langle g \mid s\rangle$ also evaluates to a stuck-term configuration with x in function position. Let l be a fresh reference, and

$$\int_x \stackrel{\mathsf{def}}{=} \begin{cases} x \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 1; \int(x)\,a\ \mathsf{else}\ \int(x)\,a \\ y \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 2; \int(y)\,a\ \mathsf{else}\ \int(y)\,a & \text{if } y \neq x \end{cases}$$

The configuration $\langle h \uplus h_{E'} \uplus l := 0 \mid E'[t]\rangle\int_x$ behaves like $\langle h \uplus h_{E'} \mid E'[t]\rangle\int$ except it sets l at 1. As a result, $\langle g \uplus h_{E'} \uplus l := 0 \mid E'[s]\rangle\int_x$ also terminates, with 1 also in l, since we can easily build a context that diverges if $!l \neq 1$. Therefore $\langle g \mid s\rangle$ does not evaluate to a value configuration, otherwise l would contain 0, and it does not evaluate to a stuck configuration with $y \neq x$ in function position, otherwise l would contain 2. As a result, there exist g', F, and w such that $\langle g \mid s\rangle \to^* \langle g' \mid F[x\,w]\rangle$.

We first show that $E\ \mathcal{R}^{\mathsf{c}}\ F$. Let h_f be a fresh store, y a fresh variable, E'' a context, $h_{E''}$ a store, $\int'$ a closing substitution, and l be a fresh reference. We define

$$\int'_x \stackrel{\mathsf{def}}{=} \begin{cases} x \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 1; \bigcup_i l_i := h_f \uplus h_{E''}; \int'(y)\ \mathsf{else}\ \int'(x)\,a \\ \\ z \mapsto \int'(z) & \text{if } z \neq x \end{cases}$$

The substitution $\int'_x$ behaves like $\int'$ except that when $\int'_x(x)$ is applied for the first time, it replaces its argument by $\int'(y)$ and sets the store to $h_f \uplus h_{E''}$. Therefore $\langle h \uplus l := 0 \mid E''[t]\rangle\int'_x \to^* \langle h_f \uplus h_{E''} \uplus l := 1 \mid E''[E[y]]\rangle\int'_x$, but this configuration then behaves like $\langle h_f \uplus h_{E''} \mid E''[E[y]]\rangle\int'$. We have the same sequence of reductions for $\langle g \uplus l := 0 \mid E''[s]\rangle\int'_x$ so we can conclude from there.

The proofs to show $v\ \mathcal{R}^{\mathsf{v}}\ w$ and $h\ \mathcal{R}^{\mathsf{h}}\ g'$ rely on similar discriminating substitution and context. The idea is to run $\langle h \uplus h_{E'} \mid E'[t]\rangle\int$, and store when $\int(x)$ is applied for the first time the value of either v or $h(l_j)$ for a given j in a fresh reference l_{v}. Because we know that the aforementioned configuration terminates, at the end of its evaluation, we can retrieve the value in l_{v} to test it. To do so, we use the same entities as in the previous case, and the extra fresh reference l_{v}. To compare v and w, we define

$$\int'_x \stackrel{\mathsf{def}}{=} \begin{cases} x \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 1; l_{\mathsf{v}} := a; \int(x)\,a \\ \qquad\qquad \mathsf{else\ if}\ !l = 1\ \mathsf{then}\ \int(x)\,a\ \mathsf{else}\ \int'(x)\,a \\ z \mapsto \lambda a.\mathsf{if}\ !l < 2\ \mathsf{then}\ \int(z)\,a\ \mathsf{else}\ \int'(z)\,a & \text{if } z \neq x \end{cases}$$

To compare h and g', we would store in l_{v} the value stored in l_j. The substitution $\int'_x$ behaves like $\int$ when $!l < 2$, and then behaves as $\int'$. It stores what should be tested when $\int'_x(x)$ is applied for the first time (when $!l = 0$). Let $E''_{l_{\mathsf{v}}} \stackrel{\mathsf{def}}{=} (\lambda z.l := 2; \bigcup_i l_i := h_f \uplus h_{E''}; E''[!l_{\mathsf{v}}\,y])\ E'$; running t in that context first evaluates $E'[t]\int$ and then behaves like $E''[!l_{\mathsf{v}}\,y]\int'$ in the proper store each time. More precisely, $\langle h \uplus h_{E'} \uplus l := 0 \mid E''_{l_{\mathsf{v}}}[t]\rangle\int'_x \to^* \langle h_f \uplus h_{E''} \uplus h_{E'} \uplus l := 2 \uplus l_{\mathsf{v}} := v \mid E''[v\,y]\rangle\int'_x$, and the resulting configuration behaves like $\langle h_f \uplus h_{E''} \mid E''[v\,y]\rangle\int'$. We have the same reductions for $\langle g \uplus h_{E'} \uplus l := 0 \mid E''_{l_{\mathsf{v}}}[s]\rangle\int'_x$ so we can conclude.

Suppose now that for all $\int$, $h_{E'}$, and E', $\langle h \uplus h_{E'} \mid E'[t] \rangle \int$ diverges. We show that $E \in \mathcal{R}{\uparrow}^{\mathsf{c}}$. Let h_f be a fresh store and y a fresh variable. Suppose there exist E', $h_{E'}$, $\int$ such that $\langle h_f \uplus h_{E'} \mid E'[E[y]] \rangle \int \Downarrow$. Let l be a fresh reference and

$$\int_x \stackrel{\mathsf{def}}{=} \begin{cases} x \mapsto \lambda a.\mathsf{if}\ !l = 0\ \mathsf{then}\ l := 1; \bigcup_i l_i := h_f \uplus h_{E'}; \int(y)\ \mathsf{else}\ \int(x)\ a \\ z \mapsto \int(z) \quad \text{if } z \neq x \end{cases}$$

Then $\langle h \uplus l := 0 \mid E'[t] \rangle \int_x \rightarrow^* \langle h_f \uplus h_{E'} \uplus l := 1 \mid E'[E[y]] \rangle \int_x$, which converges, hence a contradiction.

What remains to check are the clauses for $c \in \mathcal{R}{\uparrow}$. The case $c \rightarrow c'$ is easy to check, and the case $c = E[x\ v]$ is similar to the previous case. □

B.2 Proof of Theorem 4

We first define a notion of a compatible closure over an environment $\mathcal{E}$, for values $\mathcal{E}^{\mathsf{v}}$, terms $\mathcal{E}^{\mathsf{t}}$, evaluation contexts $\mathcal{E}^{\mathsf{c}}$, and heaps $\mathcal{E}^{\mathsf{h}}$:

$$\frac{v\,\mathcal{E}\,w}{v\,\mathcal{E}^{\mathsf{v}}\,w} \qquad \frac{}{x\,\mathcal{E}^{\mathsf{v}}\,x} \qquad \frac{t\,\mathcal{E}^{\mathsf{t}}\,s}{\lambda x.t\,\mathcal{E}^{\mathsf{v}}\,\lambda x.s}\ x \notin \mathsf{fv}(\mathcal{E})$$

$$\frac{v\,\mathcal{E}^{\mathsf{v}}\,w}{v\,\mathcal{E}^{\mathsf{t}}\,w} \qquad \frac{t_1\,\mathcal{E}^{\mathsf{t}}\,s_1 \qquad t_2\,\mathcal{E}^{\mathsf{t}}\,s_2}{t_1\ t_2\,\mathcal{E}^{\mathsf{t}}\,s_1\ s_2}$$

$$\frac{v\,\mathcal{E}^{\mathsf{v}}\,w \qquad t\,\mathcal{E}^{\mathsf{t}}\,s}{\mathsf{new}\ l := t\ \mathsf{in}\ v\,\mathcal{E}^{\mathsf{t}}\,\mathsf{new}\ l := s\ \mathsf{in}\ w}\ l \notin \mathsf{fr}(\mathcal{E}) \qquad \frac{t_1\,\mathcal{E}^{\mathsf{t}}\,s_1 \qquad t_2\,\mathcal{E}^{\mathsf{t}}\,s_2}{l := t_1; t_2\,\mathcal{E}^{\mathsf{t}}\,l := s_1; s_2}\ l \notin \mathsf{fr}(\mathcal{E})$$

$$\frac{}{!l\,\mathcal{E}^{\mathsf{t}}\,!l}\ l \notin \mathsf{fr}(\mathcal{E})$$

$$\frac{}{\square\,\mathcal{E}^{\mathsf{c}}\,\square} \qquad \frac{E\,\mathcal{E}^{\mathsf{c}}\,F \qquad t\,\mathcal{E}^{\mathsf{t}}\,s}{E\ t\,\mathcal{E}^{\mathsf{c}}\,F\ s} \qquad \frac{v\,\mathcal{E}^{\mathsf{v}}\,w \qquad E\,\mathcal{E}^{\mathsf{c}}\,F}{v\ E\,\mathcal{E}^{\mathsf{c}}\,w\ F}$$

$$\frac{E\,\mathcal{E}^{\mathsf{c}}\,F \qquad t\,\mathcal{E}^{\mathsf{t}}\,s}{l := E; t\,\mathcal{E}^{\mathsf{c}}\,l := F; s}\ l \notin \mathsf{fr}(\mathcal{E})$$

$$\frac{\mathsf{dom}(h) = \mathsf{dom}(g) \qquad \forall l \in \mathsf{dom}(h). h(l)\,\mathcal{E}^{\mathsf{v}}\,g(l)}{h\,\mathcal{E}^{\mathsf{h}}\,g}\ \mathsf{dom}(h) \cap \mathsf{fr}(\mathcal{E}) = \emptyset$$

We use the same notation for the unary counterparts of the above relations that are defined in the expected way.

Then we define a relation subst on substitutions that reify an environment $\mathcal{E}$:

$$\frac{}{\mathcal{E} \vdash \emptyset\ \mathsf{subst}\ \emptyset} \qquad \frac{\mathcal{E} \vdash \int_1\ \mathsf{subst}\ \int_2 \qquad v\,\mathcal{E}^{\mathsf{v}}\,w}{\mathcal{E} \vdash \int_1\{v/x\}\ \mathsf{subst}\ \int_2\{w/x\}}\ x \notin \mathsf{dom}(\int_1) = \mathsf{dom}(\int_2)$$

and a relation ctxt on evaluation contexts that reify a context stack Σ under an environment $\mathcal{E}$:

$$\frac{E\,\mathcal{E}^{\mathsf{c}}\,F}{\mathcal{E}, \emptyset \vdash E \;\mathsf{ctxt}\; F} \qquad\qquad \frac{E_2\,\mathcal{E}^{\mathsf{c}}\,F_2 \qquad \mathcal{E}, \Sigma \vdash E_1 \;\mathsf{ctxt}\; F_1}{\mathcal{E}, (E, F) :: \Sigma \vdash E_2[E[E_1]] \;\mathsf{ctxt}\; F_2[F[F_2]]}$$

As before we assume the corresponding unary versions of the two relations.

For each kind of simulation statement, we define a $\mathsf{closing}$ predicate stating that elements built from the environment and/or the stack are closing the simulation statement.

$$\begin{aligned}
h_0, g_0, E_0, F_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g, t, s) \text{ if } & \langle h \uplus h_0 \mid E_0[t]\rangle \textstyle\int_1 \text{ is closed},\\
& \langle g \uplus g_0 \mid F_0[s]\rangle \textstyle\int_2 \text{ is closed},\\
& \mathcal{E} \vdash \textstyle\int_1 \;\mathsf{subst}\; \int_2,\\
& \mathcal{E}, \Sigma \vdash E_0 \;\mathsf{ctxt}\; F_0,\\
& h_0\,\mathcal{E}^{\mathsf{h}}\,g_0.\\
h_0, g_0, E_0, F_0, t_0, s_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g) \text{ if } & \langle h \uplus h_0 \mid E_0[t_0]\rangle \textstyle\int_1 \text{ is closed},\\
& \langle g \uplus g_0 \mid F_0[s_0]\rangle \textstyle\int_2 \text{ is closed},\\
& \mathcal{E} \vdash \textstyle\int_1 \;\mathsf{subst}\; \int_2,\\
& \mathcal{E}, \Sigma \vdash E_0 \;\mathsf{ctxt}\; F_0,\\
& h_0\,\mathcal{E}^{\mathsf{h}}\,g_0,\\
& t_0\,\mathcal{E}^{\mathsf{t}}\,s_0.\\
h_0, E_0, \textstyle\int_0 \in \mathsf{closing}(\epsilon, \sigma, h, t) \text{ if } & \langle h \uplus h_0 \mid E_0[t]\rangle \textstyle\int_0 \text{ is closed},\\
& \mathcal{E} \vdash \textstyle\int_1 \;\mathsf{subst},\\
& \mathcal{E}, \Sigma \vdash E_0 \;\mathsf{ctxt},\\
& h_0 \in \mathcal{E}^{\mathsf{h}}.\\
h_0, E_0, \textstyle\int_0, t_0 \in \mathsf{closing}(\epsilon, \sigma, h) \text{ if } & \langle h \uplus h_0 \mid E_0[t_0]\rangle \textstyle\int_0 \text{ is closed},\\
& \mathcal{E} \vdash \textstyle\int_1 \;\mathsf{subst},\\
& \mathcal{E}, \Sigma \vdash E_0 \;\mathsf{ctxt},\\
& h_0 \in \mathcal{E}^{\mathsf{h}},\\
& t_0 \in \mathcal{E}^{\mathsf{t}}.
\end{aligned}$$

Lemma 19. *The relation $\mathcal{R}$ defined as follows is a simulation:*

$$\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2 \cup \mathcal{R}_3 \cup \mathcal{R}_4$$

where

$$\begin{aligned}
\mathcal{R}_1 = \{(\mathcal{E}, \Sigma, \langle h \mid t\rangle, \langle g \mid s\rangle) \mid \\
&\text{for all } h_0, g_0, E_0, F_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g, t, s), \\
&\text{if } \langle h \uplus h_0 \mid E_0[t]\rangle \int_1 \Downarrow \ \text{ then } \langle g \uplus g_0 \mid F_0[s]\rangle \int_2 \Downarrow\} \\
\mathcal{R}_2 = \{(\mathcal{E}, \Sigma, h, g) \mid \\
&\text{for all } h_0, g_0, E_0, F_0, t_0, s_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g), \\
&\text{if } \langle h \uplus h_0 \mid E_0[t_0]\rangle \int_1 \Downarrow \ \text{ then } \langle g \uplus g_0 \mid F_0[s_0]\rangle \int_2 \Downarrow\} \\
\mathcal{R}_3 = \{(\epsilon, \sigma, \langle h \mid t\rangle) \mid \\
&\text{for all } h_0, E_0, \int_0 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, t), \\
&\langle h \uplus h_0 \mid E_0[t]\rangle \int_0 \Uparrow\} \\
\mathcal{R}_4 = \{(\epsilon, \sigma, h) \mid \\
&\text{for all } h_0, E_0, t_0, \int_0 \in \mathsf{closing}(\mathcal{E}, \Sigma, h), \\
&\langle h \uplus h_0 \mid E_0[t_0]\rangle \int_0 \Uparrow\}
\end{aligned}$$

The proof rely on the following auxiliary lemmas.

Lemma 20. *For all h, E, t, v, x, and a fresh l, $\langle h \mid E[t]\rangle[x \mapsto v] \Downarrow$ iff $\langle h \uplus l := \lambda x.x \mid E[l := v; t]\rangle[x \mapsto \lambda y.!l\ y] \Downarrow$.*

Proof. Clear, as v is replaced by a λ-abstraction which behaves like an η-expansion of v.

Lemma 21. *For all h, E_1, E_2, v, $\int$, fresh l, and $x \in \mathsf{dom}(\int)$, $\langle h \mid E_1[E_2[v]]\rangle \int \Downarrow$ iff $\langle h \uplus l := \mathit{true} \mid E_1[x\ v]\rangle[x \mapsto \lambda y.\mathit{if}\ !l\ \mathit{then}\ l := \mathit{false}; E_2[y]\ \mathit{else}\ x\ y] \int \Downarrow$.*

Proof. The first application of $\lambda y.\mathsf{if}\ !l\ \mathsf{then}\ l := \mathsf{false}; E_2[y]\ \mathsf{else}\ x\ y]$ plugs v into E_2; any potential additional application then behaves as $\int(x)$.

Lemma 22. *Let $\mathcal{E}$, v, w, $\int'_1$, and $\int'_2$. Let $\mathcal{E}' = \mathcal{E} \cup \{(v, w)\}$. If $\mathcal{E}' \vdash \int'_1\ \mathsf{subst}\ \int'_2$ then there exist $\int_1$, $\int_2$, and a fresh variable x such that $\mathcal{E} \vdash \int_1\ \mathsf{subst}\ \int_2$, $[x \mapsto v]\int_1 = \int'_1$, and $[x \mapsto v]\int_2 = \int'_2$. Moreover we have*

1. *For all $v'\, \mathcal{E}'^{\mathsf{v}}\, w'$, there exist v'' and w'' such that $v''\, \mathcal{E}^{\mathsf{v}}\, v''$, $v' = v''[x \mapsto v]$, and $w' = w''[x \mapsto w]$.*
2. *For all $t'\, \mathcal{E}'^{\mathsf{t}}\, s'$, there exist t and s such that $t\, \mathcal{E}^{\mathsf{t}}\, s$, $t' = t[x \mapsto v]$, and $s' = s[x \mapsto w]$.*
3. *For all $h'\, \mathcal{E}'^{\mathsf{h}}\, g'$, there exist h and g' such that $h\, \mathcal{E}^{\mathsf{h}}\, g$, $h' = h[x \mapsto v]$, $g' = g[x \mapsto w]$.*
4. *For all $E'\, \mathcal{E}'^{\mathsf{c}}\, F'$, there exist E and F such that $E \in \epsilon^{\mathsf{c}}$, $E' = E[x \mapsto v]$ and $F' = F[x \mapsto w]$.*
5. *For all $\mathcal{E}', \Sigma \vdash E'\ \mathsf{ctxt}\ F'$, there exist E and F such that $\mathcal{E}, \Sigma \vdash E\ \mathsf{ctxt} F$, $E' = E[x \mapsto v]$, and $F' = F[x \mapsto w]$.*

Similarly for the unary versions of the environment, subst*, and* ctxt*.*

Proof. The lemma states that we can replace any occurrence of v and w by a fresh variable, so that we obtain entities built only from $\mathcal{E}$. The proofs are easy by induction on the derivations of the closures.

Lemma 23. *Let $h_0, E_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h, t)$ such that $\langle h \uplus h_0 \mid E_0[t]\rangle \Downarrow$. There exist h_1, g_1, E_1, F_1, $\int_1$, and $\int_2$ such that $\langle h \uplus h_0 \mid E_0[t]\rangle\int_0 = \langle h \uplus h_1 \mid E_1[t]\rangle\int_1$ and $h_1, g_1, E_0, F_1, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g, t, s)$.*

Let $h_0, E_0, t_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h)$ such that $\langle h \uplus h_0 \mid E_0[t_0]\rangle \Downarrow$. There exist h_1, g_1, E_1, F_1, t_1, s_1, $\int_1$, and $\int_2$ such that $\langle h \uplus h_0 \mid E_0[t_0]\rangle\int_0 = \langle h \uplus h_1 \mid E_1[t_1]\rangle\int_1$ and $h_1, g_1, E_0, F_1, t_1, s_1, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g)$.

Proof. In this lemma, we extend unary predicates to binary predicates. The entities h_0, E_0, and $\int_0$ may be constructed using variables and references fresh from $\pi_1(\mathcal{E})$, $\pi_1(\Sigma)$, h, t, but not from $\mathcal{E}$, Σ, g, and s. Besides, $\int_0$ may not be closing these extras entities as well. We therefore construct h_1, E_1, and $\int_1$ like h_0, E_0, and $\int_0$ are constructed, but with "fresher" variables and references when those are needed. Besides, $\int_1$ is extended so that it closes g and s.

We then build g_1, F_1, and $\int_2$ by turning the unary derivations of h_1, E_1, and $\int_1$ built out of $\pi_1(\mathcal{E})$ and $\pi_1(\Sigma)$ into binary derivations out of $\mathcal{E}$ and Σ.

With these results, we can prove Lemma 19.

Proof. **Checking $\mathcal{R}_1$:** let $\mathcal{E}, \Sigma \vdash \langle h \mid t\rangle \mathrel{\mathcal{R}} \langle g \mid s\rangle$. We proceed by case analysis on t.

Case $\langle h \mid t\rangle \to \langle h' \mid t'\rangle$. We show that $\langle h' \mid t'\rangle \mathrel{\mathcal{R}_1} \langle g \mid s\rangle$. Let $h_0, g_0, E_0, F_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h', g, t', s)$. Because $\int_1$ may not close $\langle h \mid t\rangle$, we define $\int_1'$, $\int_2'$ such that

- $\mathsf{dom}(\int_1') = \mathsf{dom}(\int_2')$;
- for $i \in \{1, 2\}$, $\int_i'(x) = \int_i(x)$ if $x \in \mathsf{dom}(\int_i)$, otherwise $\int_i'(x) = \lambda x.x$;
- $\langle h \mid t\rangle\int_1'$ is closed.

It is easy to check that $\mathcal{E} \vdash \int_1' \mathrel{\mathsf{subst}} \int_2'$, and $\langle h \mid t\rangle\int_1' \to \langle h' \mid t'\rangle\int_1$. If $\langle h' \mid t'\rangle\int_1 \Downarrow$, then $\langle h \mid t\rangle\int_1' \Downarrow$, which implies $\langle g \mid s\rangle\int_2' \Downarrow$, and therefore $\langle g \mid s\rangle\int_2 \Downarrow$, as wished.

Case $t = v$. We consider two subcases:

- Suppose $h_0, E_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h, v)$ implies $\langle h \uplus h_0 \mid E_0[v]\rangle\int_0 \Uparrow$. We show that (i) $\Sigma \neq \odot$ and (ii) $(\pi_1(\mathcal{E}) \cup \{v\}, \pi_1(\Sigma), h) \in R_4$.
  - (i) If $\Sigma = \odot$, then $\pi_1(\mathcal{E}), \odot \vdash \Box \mathrel{\mathsf{ctxt}}$. Let $\int_0$ such that $\int_0(x) = x$ for all $x \in \mathsf{fv}(h) \cup \mathsf{fv}(v)$. Then $\emptyset, \Box, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \odot, h, v)$ and since $\langle h \uplus \emptyset \mid \Box[v]\rangle\int_0 \Downarrow$, we get a contradiction.
  - (ii) Let $h_0, E_0, t_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}) \cup \{v\}, \pi_1(\Sigma), h)$. By Lemma 22, we have $h_0', E_0', t_0', \int_0'$ and x such that $h_0', E_0', t_0', \int_0'[x \mapsto v] \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h)$ and $\langle h \uplus h_0 \mid E_0[t_0]\rangle\int_0 = \langle h \uplus h_0' \mid E_0'[t_0']\rangle\int_0'[x \mapsto v\int_0']$. Then by Lemma 20, these programs diverge if and only if $\langle h \uplus \{l \mapsto \lambda x.x\} \uplus h_0' \mid E_0'[l := v; t_0']\rangle\int_0'[x \mapsto \lambda z.!l\ z]$ diverges, which is the case since $\pi_1(\mathcal{E}) \vdash \int_0'[x \mapsto \lambda z.!l\ z] \mathrel{\mathsf{subst}}$, $\{l \mapsto \lambda x.x\} \uplus h_0' \in \pi_1(\mathcal{E})^{\mathsf{h}}$ and $\pi_1(\mathcal{E}), \pi_1(\Sigma) \vdash E_0'[l := \Box; t_0'] \mathrel{\mathsf{ctxt}}$.

– Let $h_0, E_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h, v)$ such that $\langle h \uplus h_0 \mid E_0[v]\rangle \int_0 \Downarrow$. By Lemma 23, there exist $h_1, g_1, E_1, F_1, \int_1 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g, t, s)$ such that $\langle h \uplus h_1 \mid E_1[v]\rangle \int_1 \Downarrow$. By definition of $\mathcal{R}_1$, $\langle g \uplus g_1 \mid F_1[s]\rangle \int_2 \Downarrow$, so there exist w, g' such that $\langle g \mid s\rangle \to^* \langle g' \mid w\rangle$. We show that $\mathcal{E} \cup \{(v, w)\}, \Sigma \vdash h \mathrel{\mathcal{R}_2} g'$. Let $h_2, g_2, E_2, F_2, t_2, s_2, \int_3, \int_4 \in \mathsf{closing}(\mathcal{E} \cup \{(v, w)\}, \Sigma, h, g')$. By Lemma 22, there exist h'_2, g'_2, E'_2, F'_2, t'_2, s'_2, $\int'_3$, $\int'_4$ and a fresh x such that we have $h'_2, g'_2, E'_2, F'_2, t'_2, s'_2, [x \mapsto v]\int'_3, [x \mapsto w]\int'_4 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g')$ and

$$\langle h \uplus h_2 \mid E_2[t_2]\rangle \int_3 = \langle h \uplus h'_2 \mid E'_2[t'_2]\rangle [x \mapsto v]\int'_3$$
$$\langle g' \uplus g_2 \mid F_2[s_2]\rangle \int_4 = \langle g' \uplus g'_2 \mid F'_2[s'_2]\rangle [x \mapsto w]\int'_4$$

By Lemma 20, these configurations behave like respectively $\langle h \uplus h'_2 \mid E'_2[l := v; t'_2]\rangle [x \mapsto \lambda y.!l\ y]\int'_3$ and $\langle g' \uplus g'_2 \mid F'_2[l := w; s'_2]\rangle [x \mapsto \lambda y.!l\ y]\int'_4$ for a fresh l. Because $\langle g \mid s\rangle \to \langle g' \mid s'\rangle$, the last one is equivalent to $\langle g \uplus g'_2 \mid F'_2[l := s; s'_2]\rangle [x \mapsto \lambda y.!l y]\int'_4$. We conclude from there as we know that $\langle h \uplus h'_2 \mid E'_2[l := v; t'_2]\rangle [x \mapsto \lambda y.!l\ y]\int'_3 \Downarrow$ iff $\langle g \uplus g'_2 \mid F'_2[l := s; s'_2]\rangle [x \mapsto \lambda y.!l\ y]\int'_4 \Downarrow$, because $\mathcal{E}, \Sigma \vdash \langle h \mid v\rangle \mathcal{R}_1 \langle g \mid s\rangle$, $\mathcal{E} \vdash [x \mapsto \lambda y.!l\ y]\int'_3 \mathrel{\mathsf{subst}} [x \mapsto \lambda y.!l\ y]\int'_4$, $h'_2 \uplus l := \lambda x.x \mathrel{\mathcal{E}^{\mathsf{h}}} g'_2 \uplus l := \lambda x.x$, and $\mathcal{E}, \Sigma \vdash E'_2[l := \Box; t'_2] \mathrel{\mathsf{ctxt}} F'_2[l := \Box; s'_2]$.

Case $t = E[x\ v]$. We consider two subcases:

– Suppose $h_0, E_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), h, t)$ implies $\langle h \uplus h_0 \mid E_0[t]\rangle \int_0 \Uparrow$. We show that $\pi_1(\mathcal{E}) \cup \{v\}, E :: \pi_1(\Sigma) \vdash h \in \mathcal{R}\!\uparrow$.
Let $h_0, E_0, t_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}) \cup \{v\}, E :: \pi_1(\Sigma), h)$. By Lemma 22, there exist $h'_0, E'_0, t'_0, [y \mapsto v]\int'_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), E :: \pi_1(\Sigma), h)$ such that $\langle h \uplus h_0 \mid E_0[t_0]\rangle \int_0 = \langle h \uplus h'_0 \mid E'_0[t'_0]\rangle [y \mapsto v]\int'_0$. From $\pi_1(\mathcal{E}), E :: \pi_1(\Sigma) \vdash E'_0\ \mathsf{ctxt}$, we know that $E'_0 = E_1[E[E_2]]$ for some E_1 and E_2 such that $\pi_1(\mathcal{E}), \pi_1(\Sigma) \vdash E_1\ \mathsf{ctxt}$ and $E_2 \in \mathcal{E}^{\mathsf{c}}$. By Lemma 20, $\langle h \uplus h_0 \mid E_0[t_0]\rangle \int_0 \Downarrow$ iff $\langle h \uplus l := \lambda x.x \uplus h'_0 \mid E_1[E[l := v; E_2[t'_0]]]\rangle [y \mapsto \lambda z.!l\ z]\int_0 \Downarrow$ for a fresh l. By Lemma 21, the latter condition holds iff $\langle h \uplus l := \lambda x.x \uplus l :=' \mathsf{true} \uplus h'_0 \mid E_1[E[x\ v]]\rangle [x \mapsto \lambda z.\mathsf{if}\ !l'\ \mathsf{then}\ l' := \mathsf{false}; l := z; E_2[t'_0]\ \mathsf{else}\ x\ z][y \mapsto \lambda z.!l\ z]\int'_0 \Downarrow$.
We know that the last condition does not hold by the initial assumption, since one can check that $l := \lambda x.x \uplus l :=' \mathsf{true} \uplus h'_0, E_1, [x \mapsto \lambda z.\mathsf{if}\ !l'\ \mathsf{then}\ l' := \mathsf{false}; l := z; E_2[t'_0]\ \mathsf{else}\ x\ z][y \mapsto \lambda z.!l\ z]\int'_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), \mathcal{E}, t)$. Consequently, we also have $\langle h \uplus h_0 \mid E_0[t_0]\rangle \int_0 \Uparrow$, as wished.

– Let $h_0, E_0, \int_0 \in \mathsf{closing}(\pi_1(\mathcal{E}), \pi_1(\Sigma), \mathcal{E}, t)$ such that $\langle h \uplus h_0 \mid E_0[t]\rangle \int_0 \Downarrow$. Using Lemma 23 and substitutions similar to those in the proof of Theorem 2, we can show that there exist g', F, and w such that $\langle g \mid s\rangle \to^* \langle g' \mid F[x\ w]\rangle$ for some g', F, and w. We need to prove that $\mathcal{E} \cup \{(v, w)\}, (E, F) :: \Sigma \vdash h \mathrel{\mathcal{R}} g'$. We proceed like in the value case, but using the same intermediate configurations as in the previous subcase.

Checking $\mathcal{R}_2$: let $\mathcal{E}, \Sigma \vdash h \mathrel{\mathcal{R}} g$. We have to check the condition on the environment and the one on the stack.

For $\mathcal{E}$, let $v \mathrel{\mathcal{E}} w$; we prove that $\mathcal{E}, \Sigma \vdash \langle h \mid v\ x\rangle \mathrel{\mathcal{R}_1} \langle g \mid w\ x\rangle$ for a fresh x. Let $h_0, g_0, E_0, F_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g, v\ x, v\ w)$. By definition, we have $v\,x\,\mathcal{E}^{\mathsf{t}}\,w\,x$, therefore $h_0, g_0, E_0, F_0, v\,x, w\,x, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g)$. As a result, $\langle h \uplus h_0 \mid E_0[v\,x]\rangle \int_1 \Downarrow$ implies that $\langle g \uplus g_0 \mid F_0[w\,x]\rangle \int_2 \Downarrow$, which is exactly what we want.

For the stack, suppose $\Sigma = (E, F) :: \Sigma'$, and we show that $\mathcal{E}, \Sigma' \vdash \langle h \mid E[x]\rangle\ \mathcal{R}_1\ \langle g \mid F[x]\rangle$ for a fresh x. Let $h_0, g_0, E_0, F_0, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma', h, g, E[x], F[x])$. From $\mathcal{E}, \Sigma' \vdash E_0 \mathrel{\mathsf{ctxt}} F_0$ and $\square\,\mathcal{E}^{\mathsf{c}}\,\square$, we get $\mathcal{E}, \Sigma' \vdash E_0[E] \mathrel{\mathsf{ctxt}} F_0[F]$ by definition. Therefore we have $h_0, g_0, E_0[E], F_0[F], x, x, \int_1, \int_2 \in \mathsf{closing}(\mathcal{E}, \Sigma, h, g)$, so $\langle h \uplus h_0 \mid E_0[E[x]]\rangle \int_1 \Downarrow$ implies that $\langle g \uplus g_0 \mid F_0[F[x]]\rangle \int_2 \Downarrow$, which is exactly what we want.

Checking $\mathcal{R}_3$: similar to $\mathcal{R}_1$.

Checking $\mathcal{R}_4$: similar to $\mathcal{R}_2$.

$\square$

RESEARCH CENTRE
NANCY – GRAND EST

615 rue du Jardin Botanique
CS20101
54603 Villers-lès-Nancy Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399